UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Desarrollo de un sistema de análisis de imágenes médicas basado en técnicas de Deep Learning

Alicia Barrio Algarabel
Tutor: Manuel Sánchez-Montañés

Mayo 2019

Desarrollo de un sistema de análisis de imágenes médicas basado en técnicas de Deep Learning

AUTOR: Alicia Barrio Algarabel
TUTOR: Manuel Sánchez-Montañés

Grupo de Neurocomputación Biológica
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
mayo de 2019

Resumen

En este Trabajo Fin de Grado se busca realizar una comparación de las diferentes técnicas relacionadas con Deep Learning existentes en la actualidad para la implementación de un sistema de clasificación de imágenes médicas, a fin de evaluar con cuales de ellas se obtiene un mayor rendimiento.

Para ello, se comienza exponiendo un exhaustivo estudio del estado del arte sobre el campo de Deep Learning en imagen médica. Después, se seleccionan las técnicas más empleadas que se tratarán de replicar para la base de datos escogida, como las arquitecturas de la red neuronal a emplear o los distintos tipos de preprocesamiento de las imágenes del *dataset*, y se detalla en profundidad cada una de ellas. En este punto, y ante la carencia de los medios necesarios para entrenar una red neuronal profunda con nuestra base de datos, el trabajo se centra en el uso de redes pre-entrenadas para la extracción de características, técnica conocida como *transfer learning*.

Para estudiar el rendimiento de estas técnicas, se realizan una serie de experimentos en los que se va variando cada una de ellas para evaluar su impacto en la predicción final del sistema, empleando como métrica la curva ROC obtenida de cada una de las pruebas. Tras ello se realiza una comparativa de las técnicas escogidas, aportando una serie de recomendaciones a seguir a la hora de desarrollar un sistema de estas características derivadas de estos resultados obtenidos.

Por último, se recogen las conclusiones de este trabajo y se comenta el posible trabajo futuro en relación con el tema escogido.

Palabras clave (castellano)

Deep Learning, *Machine Learning*, red neuronal, imagen médica, *transfer learning*.

Abstract

The aim of this Thesis is to make a comparison of the different Deep Learning techniques used nowadays for the implementation of a medical imaging classification system in order to evaluate which of them provide a better performance.

To achieve this, we begin by presenting an exhaustive state of the art study on the field of Deep Learning in medical imaging. Then, the most used techniques which will be tried to be replicated for our dataset are chosen, such as the neural network architecture or the different types of preprocessing for the dataset images, and each of them is deeply described. At this point, in absence of the necessary means to train a deep neural network with our database, the work focuses on the use of pre-trained networks for the feature extraction, a technique known as transfer learning.

To study the performance of these techniques, some experiments are carried out in which each parameter is modified to evaluate its impact on the final prediction of the system, using as a metric the ROC curve obtained from each of the tests. After that, a comparison of the chosen techniques is made, providing some recommendations, extracted from the obtained results, to follow when developing a system of these characteristics.

Finally, the conclusions of this work are exposed, and the possible future work related to the chosen topic is discussed.

Keywords

Deep Learning, Machine Learning, neural network, medical imaging, transfer learning.

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor, Manuel, por la ayuda recibida durante el desarrollo de este trabajo.

También quiero dar las gracias a todas las personas que han estado a mi lado durante la carrera, que han confiado en mi desde el principio y han estado ahí en los malos momentos, pero también han hecho de estos años una de las mejores etapas de mi vida. Gracias a mi familia, amigas y amigos, y toda la gente maravillosa que he conocido durante estos años de universidad.

Por último, agradecer a esas tres personas que conocí allá por septiembre de 2014 y que hoy puedo decir que son mis amigas, con las que he compartido muchas horas de clase, noches de fiesta, viajes por toda Europa y sobre todo mucho apoyo y cariño siempre. Pau, Julia, Marta: sois lo mejor que me llevo de esta carrera.

Y en especial gracias a Paula, por estar conmigo desde el primer día hasta el último, por aguantarme, ayudarme, y hacer esta etapa tan dura muchísimo más fácil. Gracias por demostrar que, aunque ya no estemos todo el día juntas, nada ha cambiado.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

En los últimos años, el Machine Learning y en concreto el Deep Learning están demostrando una gran capacidad para resolver todo tipo de problemas, especialmente en el área de la Visión Artificial [Litjens et al., 2017]. En este contexto se presenta una magnífica oportunidad para aplicar dicha tecnología al campo de la imagen médica. Este tipo de imágenes, que se suelen adquirir mediante técnicas como los Rayos-X o la Resonancia Magnética (RM), permiten realizar un estudio acerca del estado de los huesos, los tejidos o los distintos órganos de los pacientes, pero requieren que este se lleve a cabo por parte de un profesional [Shen et al., 2017]. Si los equipos clínicos dispusiesen de software capaz de llevar a cabo estas tareas, la robustez de los estudios realizados por los mismos aumentaría considerablemente, lo cual repercutiría directamente en la calidad de los servicios médicos.

En este trabajo se aborda una de las tareas básicas dentro de la imagen médica: la clasificación. Esta consiste, a grandes rasgos, en decidir si en una imagen de cierto órgano o tejido del cuerpo está o no presente una determinada lesión o enfermedad. Se trata de una tarea que puede resultar relativamente sencilla para los seres humanos, especialmente para médicos expertos, pero va a presentar una alta complejidad a la hora de automatizarla. El hecho de poder ser realizada mediante máquinas supondría que estas pudieran asistir a los expertos encargados de llevarla a cabo, o incluso en un futuro sustituir la necesidad de que los médicos tengan que realizarla, reduciendo drásticamente su carga de trabajo y obteniendo diagnósticos más rápidos, lo que en muchos casos podría beneficiar al paciente.

Son múltiples las publicaciones realizadas en el campo, como Gulshan et al, 2016 y Esteva et al, 2017, algunas de ellas expuestas a lo largo de este trabajo, que reseñan el potencial del Deep Learning para esta aplicación, ya que en muchos casos se llega a alcanzar niveles de robustez similares a los del análisis de expertos humanos. Sin embargo, actualmente no se cuenta con mucha experiencia en la implantación de este tipo de tecnología en sistemas de asistencia médica reales, quizá debido a que esta requiere recursos de los que los centros médicos aun no disponen.

Por todo ello, la motivación de este trabajo es poner de manifiesto el gran potencial que presenta el Deep Learning en el campo de la medicina, exponiendo y comparando las distintas técnicas halladas mediante el estudio del estado del arte que aumentan la viabilidad de su aplicación real, mediante la utilización de equipos estándar con los que ya se cuenta en la mayoría de los centros médicos.

1.2 Objetivos

El principal objetivo de este trabajo es desarrollar un sistema de clasificación de imagen médica que presente niveles de robustez similares a los hallados mediante el estudio del estado del arte utilizando un equipo estándar similar al que suele tener un usuario medio. Para ello:

- Se realiza un estudio del estado del arte en profundidad para conocer los métodos aplicados hasta el momento para la resolución del problema propuesto.
- Se seleccionan aquellos que presentan un mejor rendimiento de entre los que su aplicación resulta viable con los medios de los que se dispone.
- Se aplican las diferentes técnicas finalmente seleccionadas a la base de datos elegida para realizar los experimentos.
- Se lleva a cabo una comparación del rendimiento de los diferentes métodos seleccionados con el objetivo de exponer cuales de ellos resultan de mayor o menor ayuda a la hora de enfrentarse a un problema de clasificación de imagen médica.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo 1: Introducción.** Se exponen los principales motivos que han impulsado la realización de este trabajo y los objetivos que se pretenden alcanzar con él.
- **Capítulo 2: Estado del Arte.** Se presenta una pequeña introducción al Deep Learning y la aplicación de este a imagen médica, así como se aborda una extensa exposición de diversas publicaciones realizadas acerca del tema.
- **Capítulo 3: Metodología.** Se describe en profundidad la base de datos empleada, así como los distintos preprocesamientos que se le aplican, las distintas arquitecturas y técnicas seleccionadas y el software utilizado para todo ello.
- **Capítulo 4: Integración, pruebas y resultados.** Se explican y comparan los resultados de cada uno de los experimentos llevados a cabo.
- **Capítulo 5: Conclusiones y trabajo futuro.** Se comentan las conclusiones extraídas tras la realización de este trabajo y las posibles líneas de trabajo futuro planteadas.

2 Estado del arte

En este capítulo se proporciona una pequeña introducción al Deep Learning y la aplicación de este a la imagen médica. Se dividirá en cinco secciones, en la primera de ellas se realizará un breve repaso de la evolución del análisis automático de imagen a modo de introducción, en la segunda, tercera y cuarta se describirán los principales aspectos del Deep Learning y las redes neuronales, mientras que en la quinta se hará lo propio con el campo en el que se ha desarrollado este trabajo, las aplicaciones del Deep Learning en medicina.

2.1 Introducción

En los inicios del análisis automático de imagen, entre las décadas de 1970 y 1990, este se realizó mediante el procesamiento secuencial de píxeles de bajo nivel, aplicando filtrados de detección de bordes y líneas, dilatación de regiones, etc. construyendo así sistemas basados en reglas capaces de resolver problemas particulares. A finales de los noventa, los sistemas completamente diseñados por seres humanos comenzaron a dar paso a sistemas entrenados por ordenadores. Para ello, se utilizaron las denominadas técnicas supervisadas, las cuales eran capaces de extraer vectores de características de los datos de entrenamiento [Litjens et al., 2017]. Un proceso muy importante en el diseño de tales sistemas era el de la selección de características que representaran óptimamente los datos para cada problema, proceso que aún era realizado por personas.

El siguiente paso fue conseguir que los ordenadores aprendiesen por sí mismos estas características discriminantes de las imágenes. Los algoritmos de Deep Learning están basados en este concepto: son redes compuestas por una serie de capas cuya función es transformar las entradas en salidas, mediante el aprendizaje de características de mayor nivel cada vez en las capas intermedias [Shen et al., 2017]. La arquitectura más utilizada actualmente para el análisis de imágenes han sido las Redes Neuronales Convolucionales (Convolutional Neural Networks), de ahora en adelante CNNs.

Las CNNs son redes formadas por muchas capas que transforman su entrada en una nueva señal de salida mediante filtros de convolución. Estas redes aprenden a extraer información relevante y robusta de las imágenes. Esto significa que la red debe ser invariante frente a desplazamientos, giros, iluminación, tamaño, etc. El desarrollo de CNNs se ha venido realizando desde finales de la década de los setenta, pero teniendo un uso menor hasta que se pudo entrenar de manera eficiente estas redes profundas. La transición a la utilización de este tipo de redes ha sido gradual, pero hubo un punto de inflexión tras la contribución de Krizhevsky et al., 2012 al *challenge* de ImageNet en diciembre de 2012, con su red llamada AlexNet. En los años siguientes, se han ido desarrollando arquitecturas más profundas, pero basadas en esta. Actualmente, las redes convolucionales profundas son la técnica más utilizada en visión artificial, y, por tanto, en los sistemas de análisis de imagen médica [Litjens et al., 2017].

2.2 Deep Learning: contexto general

En esta sección se van a tratar los principales aspectos del Deep Learning. El número de publicaciones sobre el tema, y no solo de aplicaciones en imagen médica, sino en todos los campos, ha tenido un importante crecimiento en los últimos años. Actualmente, el Deep Learning y, concretamente, las redes convolucionales profundas se han convertido en la técnica más utilizada en visión artificial. En general, los problemas a resolver con este tipo de redes son de aprendizaje supervisado, es decir, problemas en los que se tiene un conjunto de datos de características de entrada '*x*' y sus pares o etiquetas '*y*', las cuales representan una instancia de un conjunto fijo de clases. Aunque existen gran cantidad de matices, se pueden clasificar en dos tipos: tareas de clasificación, donde el número de clases será discreto, y de regresión, donde este será un vector de valores continuos [Litjens et al., 2017].

Los algoritmos de Machine Learning tradicionales necesitan, para su funcionamiento, la aplicación previa de *'feature engineering'*, que es el proceso de extracción de las características discriminantes de la imagen, y tiene que ser llevado a cabo por humanos [Shen et al., 2017]. Esto da como resultado algoritmos diseñados para detectar objetos o características concretas en la imagen, y además es particularmente complicado y costoso. El Deep Learning es una técnica de Machine Learning especialmente eficaz en la detección automática de características, por lo que evita dicho *feature engineering*. Esta técnica implica la alimentación del modelo con grandes volúmenes de datos, lo cual puede suponer un problema. Se trata de algoritmos capaces de aprender sin intervención humana previa, sacando por sí mismos conclusiones acerca de la semántica embebida en los datos, es decir, son capaces de aprender representaciones de datos.

El Deep Learning se basa en estructuras lógicas inspiradas en la organización del sistema nervioso de los mamíferos. Está formado por capas de unidades de proceso, denominadas neuronas artificiales, que se especializan en detectar determinadas características existentes en la señal que les llega como entrada. Cada una de estas capas transforma su entrada en una representación más abstracta, por ejemplo, en el caso de imágenes las primeras capas detectan bordes, contornos, después formas sencillas, posteriormente formas más complejas, etc [Shen et al., 2017].

Su objetivo es la creación de sistemas cognitivos artificiales, capaces de tomar decisiones y realizar millones de tareas diferentes que solo eran capaces de llevar a cabo los humanos: Leer páginas web con una excelente comprensión lectora, reconocer los rostros que aparecen en las imágenes publicadas en redes sociales [Parkhi et al., 2015], comprender la emoción contenida en el tono de voz de una conversación telefónica [Devillers et al., 2005], contestar a las preguntas de un cliente en un chat, entender la dinámica y los motivos de los movimientos geográficos de las personas, predecir el gasto energético de una fábrica, inferir qué películas o canciones gustarán más a cada persona [Van den Oord et al., 2013], recomendar la dieta y el ejercicio más saludable para cada persona en función de su estado actual de salud y su genotipo, y muchísimas aplicaciones más ya existentes o que potencialmente se pueden desarrollar.

2.3 Redes neuronales estrechas

2.3.1 Elementos básicos

Las redes neuronales son un tipo de algoritmo de aprendizaje automático inspirado en cómo funciona el sistema nervioso de los animales (bioinspiradas), que constituye la base de la mayoría de los métodos de Deep Learning. Se trata de un sistema de interconexión de neuronas que colaboran entre sí para producir un estímulo de salida [LeCun et al., 2015].

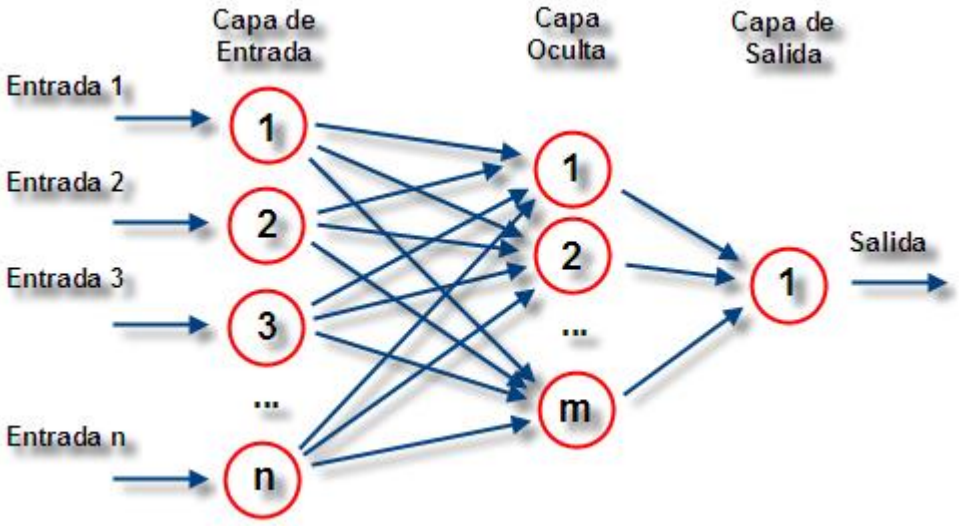


Figura 2-1: Esquema de una red neuronal [fuente]

Al igual que en una red neuronal biológica, en una red neuronal artificial la neurona es el elemento básico. En ella las dendritas reciben señales de entrada y el cuerpo celular las combina e integra emitiendo señales de salida a través del axón. Además, las conexiones entre neuronas se llevan a cabo mediante la sinapsis.

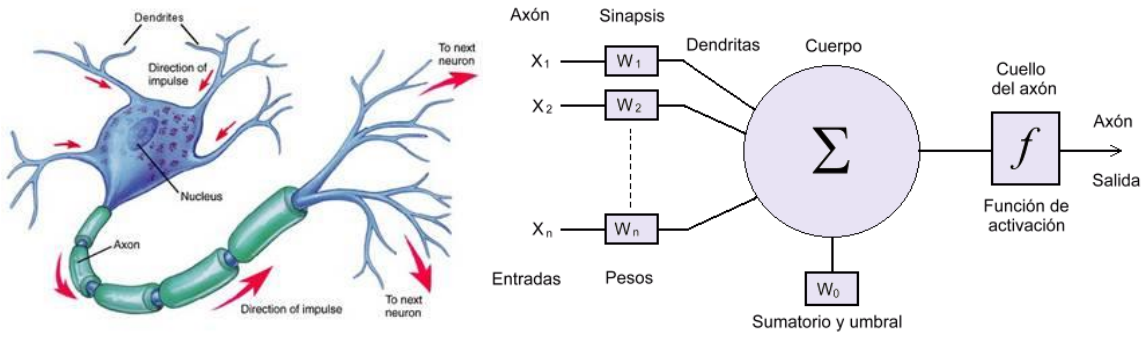


Figura 2-2: Neurona biológica y neurona artificial [fuente] [fuente]

Una red neuronal artificial es un sistema formado por neuronas interconectadas y organizadas jerárquicamente. La información se procesa a través de los estados de las neuronas como respuesta a entradas externas. Estas neuronas se organizan en capas, el flujo de información es direccional, y en ellas la sinapsis serían los pesos de las conexiones. Cada neurona recibe información de las conexiones con otras neuronas, la procesa, y envía el resultado a otras neuronas. En la aproximación más sencilla este procesamiento de información consiste en realizar una suma pesada de las informaciones y pasar esta suma por una función no lineal denominada 'función de activación', es decir, $y = f\left(\sum_{i=0}^{i=n} wi * xi\right)$, donde *y* es la salida de la neurona, *f* sería la función de activación, *wi* el peso asociado a cada entrada, y *xi* la entrada de la neurona. Se suelen emplear los siguientes tipos de función de activación: [Xin Yao, 1999]

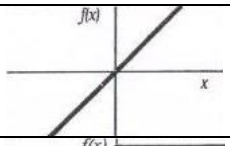
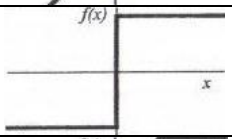
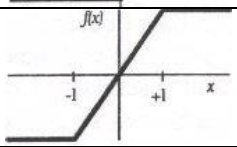
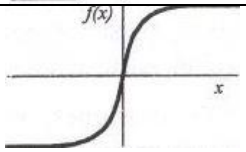
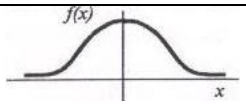
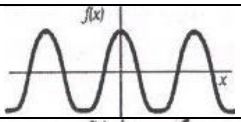
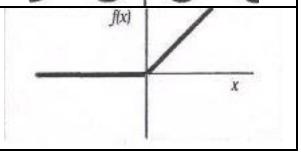
	Función	Rango	Gráfica
Identidad	$y = x$	[-∞, +∞]	
Escalón	$y = \text{sign}(x)$ $y = H(x)$	{-1, +1} {0, +1}	
Lineal a tramos	$y = \begin{cases} -1, si\ x < -l \\ x, si\ +l \leq x \leq l \\ +1, si\ x > +l \end{cases}$	[-1, +1]	
Sigmoidea	$y = \frac{1}{1+e^{-x}}$ $y = \text{tgh}(x)$	[0, +1] [-1, +1]	
Gaussiana	$y = Ae^{-Bx^2}$	[0, +1]	
Sinusoidal	$y = A\text{sen}(\omega x + \varphi)$	[-1, +1]	
ReLU	$y = max(0,x)$	[-∞, +∞]	

Tabla 2-1: Funciones de activación comunes

2.3.2 El perceptrón multicapa

Los perceptrones multicapa (MLP), popularizados a finales de los años 80, son uno de los tipos de redes neuronales tradicionales más conocidos. Se basan en una capa de entrada, una o varias capas medias, normalmente denominadas capas ocultas, y finalmente una capa de salida. En ellas el flujo de información va de izquierda a derecha, es decir, la entrada se

propaga hacia delante (*feedforward*). Algunas de las ventajas aportadas por los MLPs son el aprendizaje adaptativo, autoorganización, tolerancia a fallos o flexibilidad. Además, se trata de un aproximador universal, es decir, que cualquier función puede aproximarse con el grado de precisión deseado con un MLP con al menos una capa oculta de neuronas, pero cuantas más capas tenga la red, más potente será esta [Shen et al., 2017].

Para su entrenamiento se utiliza una base de datos con ejemplos etiquetados, es decir, se trata de aprendizaje supervisado. Después, se fija cómo debe calcularse el error cometido por la red cuando su salida no es la deseada, lo cual dependerá del tipo de problema (regresión, clasificación…), de cómo se codifiquen las salidas, y de si algún tipo de error debe penalizarse más que otros. El algoritmo inicializa los pesos de forma aleatoria y posteriormente busca de manera iterativa los valores para los pesos de las conexiones que minimizan este error. Es decir, se trata de un proceso de aprendizaje iterativo, ya que cada peso se actualiza de acuerdo con su responsabilidad en el error, tras esto se calcula el error cometido con los nuevos pesos mediante el algoritmo *backpropagation* (propagación hacia atrás), y así sucesivamente. La fórmula concreta utilizada para actualizar los pesos se deriva del gradiente del error respecto a los pesos. Este algortimo de propagación hacia atrás diluye el gradiente de forma exponencial a medida que atraviesa capas en su camino hasta el principio de la red, por lo que en una red muy profunda (con muchas capas ocultas), sólo las últimas capas se entrenan, mientras que las primeras apenas sufren cambios. Por ello, compensa utilizar redes con pocas capas ocultas que contengan muchas neuronas en lugar de redes con muchas capas ocultas que contengan pocas neuronas [Shen et al., 2017].

Esto era así hasta la llegada de las técnicas de Deep Learning.

2.4 Redes neuronales convolucionales profundas

La principal diferencia que existe entre las CNNs y las redes neuronales tradicionales como los MLPs reside en la cantidad de capas ocultas que poseen: una red neuronal básica puede tener de dos a tres capas, mientras que una red de aprendizaje profundo puede tener decenas o centenas. En Deep Learning, redes neuronales con muchas capas ocultas pueden ser bien entrenadas si los pesos son inicializados de una forma inteligente y no de forma aleatoria. Además, en este tipo de arquitecturas profundas las funciones de activación que más se utilizan son *Rectified Linear Unit* (ReLU), y *Softplus* [LeCun et al., 2015].

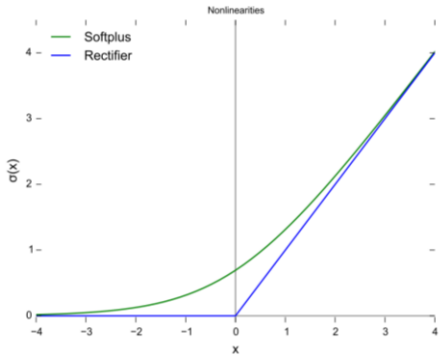


Figura 2-3: Funciones *Rectified Linear Unit* (ReLU), y *Softplus* [fuente]

En este tipo de redes neuronales se realizan operaciones de convolución y *max-pooling* sobre las imágenes de entrada. Mediante estos procesos son capaces de extraer información relevante de los datos de entrada, es decir, en el caso de imágenes, se comportan de manera robusta frente a desplazamientos, giros, iluminación, tamaño, etc., y, además, se reduce drásticamente la cantidad de parámetros en las mismas, porque el número de pesos ya no va a depender del tamaño de las imágenes de entrada [Shen et al., 2017].

2.4.1 Elementos de una CNN

En primer lugar, se divide la imagen de entrada en distintos *patches* solapados. Cada neurona de la capa siguiente se encarga de procesar la información de cada *patch*.

En las capas convolucionales la señal de entrada se va transformando de forma continua en una nueva señal de salida, concretamente, en Deep Learning se suelen realizar convoluciones lineales, que se hacen con filtros convolutivos. Mediante estos filtros se pueden realizar detectores de bordes o esquinas para imágenes, correctores de ruido, detectores de movimiento en vídeo, y muchas otras aplicaciones. Además, estos tienen asociada una matriz de pesos cuadrada denominada *kernel* cuyas dimensiones son normalmente impares e iguales y mucho menores que la información de entrada, por lo que el tamaño de salida será igual o menor que el de entrada y contendrá las características más relevantes extraídas de la misma. La posición del píxel central del *kernel* se va a corresponder con la posición del píxel de salida. Un ejemplo de esto serían los filtros de realce que resaltan los detalles más finos de la imagen, o las máscaras de convolución en las que los coeficientes suman un total de 1, como los filtros paso bajo usados en el suavizado de imagen [Krizhevsky et al., 2012].

Entre las distintas capas convolucionales suelen insertarse capas de *max-pooling*, cuyo fin es el de reducir progresivamente el tamaño de la imagen submuestreándola, disminuyendo así la cantidad de parámetros, lo cual también ayuda a controlar el sobreajuste. En ella, los valores de los píxeles vecinos se agregan usando una función de permutación invariante, típicamente la operación máximo o media. En imágenes esto añade robustez a traslaciones [Murray et al., 2014].

Finalmente, tras la secuencia de capas convolucionales de la red, se añaden capas '*fully connected*' (completamente conectadas), que son las capas típicamente empleadas en redes neuronales tradicionales, donde cada una de las neuronas tiene conexiones completas a todas las activaciones de la capa anterior [Schwing et al., 2015].

En la siguiente figura se puede observar una red neuronal 'tradicional' y una convolucional para un problema de clasificación de 10 categorías en el que se utilizan imágenes de 32x32 en RGB. A la izquierda una red neuronal regular de 3 capas. A la derecha una red neuronal convolucional que organiza sus neuronas en tres dimensiones (ancho, alto, profundidad). Cada capa transforma el volumen de entrada 3D a un volumen de salida 3D de activaciones neuronales. En este ejemplo, la capa de entrada (roja) contiene la imagen, por lo que sus dimensiones serían las mismas que las de esta, en este caso 32x32x3, y la última capa (azul) tendría dimensiones 1x1x10, es decir, un valor por cada una de las categorías [CS231n Convolutional Neural Networks for Visual Recognition].

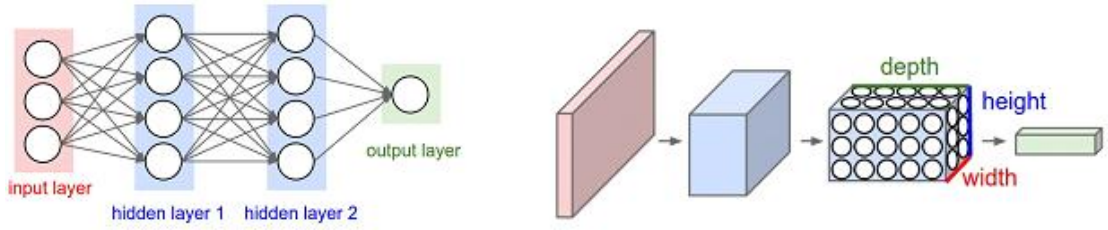


Figura 2-4: Esquema de una red neuronal 'tradicional' y una red neuronal convolucional [fuente]

2.4.2 Entrenamiento de una CNN

Al igual que en las redes neuronales tradicionales, el algoritmo de entrenamiento estándar en estas redes consiste en actualizar los pesos mediante descenso por gradiente, propagando el error hacia atrás en la red. Se suelen utilizar distintas variantes, como el SGD (*Stochastic Gradient Descent*), Adam o RMSProp [Ruder S., 2016].

Originalmente, la estrategia de actualizar los pesos computando el gradiente de la función de coste para cada dato de entrenamiento se denominaba 'Stochastic Gradient Descent', mientras que cuando esto se hacía para todo el conjunto de entrenamiento se llamaba 'batch gradient descent'. En las librerías actuales es posible fijar para cualquier algoritmo de descenso por gradiente la cantidad de datos que se utilizan para calcular el gradiente de la función objetivo (el tamaño del 'minibatch'). La idea es usar un tamaño que no sea ni muy pequeño ni muy grande de tal forma que al realizar actualizaciones en los pesos para cada mini lote de n ejemplos el descenso por gradiente converja rápido, pero de manera estable. Este parámetro deberá seleccionarse manteniendo un compromiso entre la precisión de la actualización de los pesos y el tiempo que lleva realizar dicha actualización. Los tamaños de mini lotes comunes varían entre 50 y 256 datos de entrenamiento, pero pueden ser distintos para las diferentes aplicaciones.

Otro parámetro a fijar es el número de veces que el algoritmo de aprendizaje se aplica al conjunto de datos de entrenamiento. Cada vez que esto ocurre para todo el conjunto de entrenamiento se denomina *epoch* (época). Normalmente se suele fijar en varias decenas, pero también se puede seleccionar de forma automática detectando cuando la precisión de la clasificación ha dejado de mejorar [Ruder S., 2016].

La complejidad de las CNN es muy elevada ya que cuentan con una enorme cantidad de parámetros que entrenar, por lo que el algoritmo de aprendizaje puede quedar ajustado a unas características muy específicas de los datos de entrenamiento, lo que se conoce como sobreajuste (*overfitting*) [Shen et al., 2017]. Para que esto no ocurra el volumen de datos de entrenamiento debe ser muy elevado, lo cual no siempre es posible debido al tamaño limitado de las bases de datos disponibles. Por ello, existen ciertas técnicas para tratar de evitar el sobreajuste sin tener que utilizar cantidades de datos tan enormes.

2.4.3 Técnicas para controlar el sobreajuste

2.4.3.1 Data augmentation

El *data augmentation* es uno de los métodos más comúnmente empleados para reducir el sobreajuste en las CNN y además ayuda a mejorar su rendimiento en problemas con clases desbalanceadas. En el caso del análisis automático de imagen, consiste en aumentar artificialmente el dataset realizando transformaciones de distintos tipos en las imágenes [LeCun et al., 2015]. Varios *papers* que aplican Deep Learning a imagen médica emplean este tipo de estrategias ya que hacen que las redes sean más robustas y obtengan un buen rendimiento.

En la tarea de clasificación de imágenes de retina se utilizan diversos tipos de *data augmentation*. En primer lugar, los procedimientos más utilizados son la rotación de las imágenes y voltearlas/reflejarlas vertical u horizontalmente. Por ejemplo, realizar rotaciones aleatorias entre 0° y 90° y reflejar o no vertical y horizontalmente las imágenes al azar [Pratt et al., 2016], o rotar las imágenes aleatoriamente entre 0° y 360° [Graham, 2015]. En otros casos realizan estas operaciones pero no con la imagen original completa sino con parte de ella, como reflejar horizontal y verticalmente parches aleatorios de 224x224 de imágenes cuyo tamaño original es 256x256 [Chen et al., 2015], o rotar y voltear trozos al azar del 96% de las imágenes originales [Worral et al., 2016].

Otra forma de aplicar *data augmentation* consiste en alterar las intensidades de los canales RGB en las imágenes de entrenamiento [Antony et al., 2015]. Específicamente, realizan PCA en el conjunto de valores de píxeles RGB en todo el conjunto de entrenamiento. A cada imagen de entrenamiento, se le agregan múltiplos de los componentes principales, con magnitudes proporcionales a los valores propios correspondientes por una variable aleatoria gaussiana con media cero y desviación estándar 0.1. Esta aproximación captura una propiedad importante de las imágenes naturales, que se trata de que la identidad del objeto es invariable a los cambios en la intensidad y el color de la iluminación. Por otra parte, se realizan otras operaciones como traslaciones, shift [Pratt et al., 2016], escalados o skew [Graham, 2015].

2.4.3.2 Dropout

El término *dropout* se refiere a eliminar neuronas, junto con todas sus conexiones entrantes y salientes, de manera temporal en una red neuronal. Esto se lleva a cabo de forma aleatoria, es decir, en el caso más simple, se pone a 0 una unidad con una probabilidad fija p independiente de otras neuronas, donde p puede elegirse utilizando un conjunto de validación o simplemente puede establecerse en 0.5, valor que parece óptimo para un gran número de arquitecturas y tareas distintas [Srivastava et al., 2014]. De esta manera no contribuyen en la propagación hacia delante de la entrada y no cuentan en el cálculo del error mediante propagación hacia atrás. Por lo tanto, para cada nueva entrada la red neuronal muestrea una arquitectura diferente, pero todas ellas a comparten pesos. Se trata de una técnica que se podría describir como una manera eficiente de combinar las

predicciones de distintos modelos, lo cual mejora el rendimiento del algoritmo [Srivastava et al., 2014].

En la mayoría de la literatura consultada sobre modelos para la clasificación de imágenes de retina se emplea esta técnica con la finalidad de reducir el sobreajuste. En concreto, se suele utilizar tras las capas densas del final de la red, con una probabilidad *p* de 0.5 [Pratt et al., 2016] [Chen et al., 2015] [Graham, 2015]. También hay casos de redes que lo hacen con una probabilidad de 0.1 [Graham, 2015].

2.4.3.3 Batch normalization

Se trata de un tipo de normalización que corrige las medias y las varianzas de las entradas a las capas. Como su nombre indica, esto se hace en cada *batch*, es decir, en cada 'lote' de imágenes en el que se divide el conjunto de datos en cada *epoch*. Principalmente se aplica para acelerar el entrenamiento de las redes, y además regulariza el modelo de forma que se necesita una menor utilización de *dropout* [Ioffe et al., 2015].

En la literatura consultada sobre modelos para la clasificación de imágenes de retina se encuentran varios ejemplos donde se aplica esta técnica [Pratt et al., 2016] [Gulshan et al, 2016].

2.4.3.4 Regularización

La regularización consiste en introducir una penalización por la complejidad del modelo en la función de coste, de forma que se minimiza la magnitud de los pesos. De esta forma se previene el *overfitting*, evitando que la red se ajuste demasiado al conjunto de entrenamiento. Existen varios tipos de regularización. El más común es conocido como regularización L2, donde este término de penalización es la suma de los cuadrados de los pesos. También existen otros como la regularización L1 donde el término de penalización se calcula mediante el valor absoluto de los pesos [Zhang et al., 2016].

En algunas de las redes para la clasificación de imágenes de retina estudiadas emplean esta técnica, en concreto la regularización L2, a fin de reducir el sobreajuste [Pratt et al., 2016] [Antony et al., 2015].

2.4.3.1 Transfer learning

Es, esencialmente, el uso de redes previamente entrenadas en bases de datos de gran tamaño relativas a otros problemas, cuyos pesos aprendidos se utilizan para inicializar la red a la hora de entrenarla para el problema concreto para el que se va a utilizar. Se puede usar una red pre-entrenada como extractor de características o entrenar alguna capa de una red pre-entrenada en los datos/imágenes del problema concreto para el que se vaya a utilizar, esto último es conocido como *fine-tuning* [LeCun et al., 2015].

Se trata de una estrategia muy utilizada en los problemas de clasificación de imágenes médicas, donde no se suele contar con grandes bases de datos con los que entrenar las redes. Posteriormente se comentan varios ejemplos de ello.

2.4.4 Arquitecturas

Tras el estudio de la literatura se ha llevado a cabo una división en varios grupos de los distintos tipos de arquitecturas que se emplean en los problemas de clasificación de imagen médica y en concreto en aquellos trabajos dedicados a la detección de retinopatía diabética. Estos se detallan en los siguientes apartados.

2.4.4.1 Redes entrenadas desde cero

En algunos de los trabajos analizados utilizan una única CNN que entrenan por completo en las imágenes de su base de datos, por ejemplo, para la clasificación de retinopatía diabética (DR) en imágenes de fondo de ojo con la siguiente arquitectura: varios bloques de convolución con activación *leaky* ReLU (cuya diferencia con la función ReLU se puede observar en la figura 2-5) con valor 0.01 y *batch normalization* después de cada capa. A medida que aumenta el número de mapas de características, solo se realiza *batch normalization* por bloque. El *max-pooling* se realiza con *kernels* 3x3 y *stride* 2x2. Tras el último bloque convolucional, la red se aplana a una dimensión. Después, hay 2 capas densas con *dropout* (0.5) para reducir el sobreajuste, y una capa densa final con 5 neuronas que utiliza un clasificador *softmax* para predecir la clasificación. Además, para disminuir el sobreajuste se utilizan pesos distintos para cada clase relativos la cantidad de imágenes que hay de cada una de ellas. Por otra parte, usan regularización L2 en las capas convolucionales. Los pesos se inicializan aleatoriamente siguiendo una distribución Gaussiana. Utilizan un tamaño de imagen de entrada de 512×512×3 píxeles [Pratt et al., 2016].

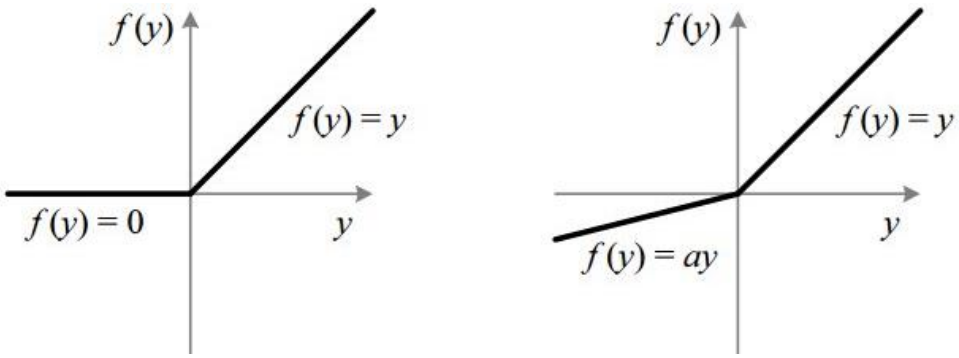


Figura 2-5: Diferencia entre Rectified Linear Unit y Leaky Rectified Linear Unit [fuente]

En otro ejemplo, para abordar el problema de la clasificación de glaucoma en imágenes de fondo de ojo entrenan una red compuesta por 6 capas, las 4 primeras son capas convolucionales con ReLUs y las dos últimas son completamente conectadas. La salida de la última de ellas alimenta un clasificador *softmax* para la predicción de la probabilidad de

presencia de glaucoma en las imágenes. La entrada a la red son imágenes de 3 canales y 256×256 píxeles. Para evitar el sobreajuste, tras las dos primeras capas convolucionales, hay dos capas de normalización de respuesta. Además, también para reducir el sobreajuste, estas capas de normalización de respuesta y todas las convolucionales van seguidas de capas de *max-pooling* con *overlapping* (superpuestas). Para mejorar el rendimiento del algoritmo se utiliza *data augmentation* consistente en traslaciones y reflexiones horizontales de las imágenes, y *dropout* con probabilidad de 0.5 en las capas densas solo durante el entrenamiento, para las imágenes de test se usan todas las neuronas [Chen et al., 2015].

Para poder entrenar desde cero redes tan complejas, se aplican otras estrategias como entrenar previamente redes menores con imágenes más pequeñas para inicializar los pesos de redes de mayor tamaño. En otro trabajo, para la clasificación de retinopatía diabética en imágenes de retina, usan dos redes neuronales formadas por las mismas capas, pero con distintos tamaños de filtro. Consisten en 13 capas convolucionales, 4 de ellas seguidas de una capa de *max-pooling*, y una de *RMSpool* (que también submuestrea pero haciendo la media cuadrática en vez del máximo de una región), y finalmente capas densas seguidas de *dropout* para disminuir el sobreajuste. Todas las capas convolucionales y densas usan función de activación *leaky* ReLU y regularización L2 con factor 0.0005. Para poder entrenar estas grandes redes convolucionales desde cero previamente entrenan redes más pequeñas en imágenes de 128×128 píxeles y usan los pesos entrenados para inicializar (parcialmente) redes de tamaño intermedio que se entrenan en imágenes de 256×256 píxeles. Después, repiten este procedimiento para las redes finales que son entrenadas con imágenes de 512×512 píxeles [Antony et al., 2015].

Por tanto, de los modelos entrenados desde cero estudiados se puede concluir que en general entrenan redes formadas por varias capas convolucionales con ReLU/leaky ReLU, varias densas con *dropout* (0.5) para reducir el sobreajuste y finalmente un clasificador *softmax*. Las capas de *max-pooling* superpuestas también mejoran el rendimiento. Además, se emplean técnicas como el *data augmentation* para reducir el sobreajuste. Para inicializar los pesos lo hacen de forma aleatoria (distribución Gaussiana) y usan regularización L2 para su entrenamiento. Por otro lado, las dimensiones de la capa de entrada varían entre 512×512×3 y 256×256×3 píxeles.

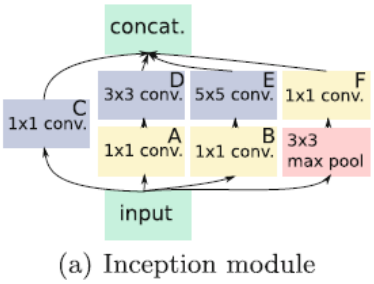
2.4.4.2 Utilización de redes pre-entrenadas

En otras muchas publicaciones se puede observar que utilizan la técnica del *transfer learning* para este problema de clasificación.

Un buen ejemplo es la utilización de la CNN Inception-v3 de GoogLeNet pre-entrenada con la base de datos de ImageNet para la clasificación de lesiones de la piel. Después esta red se entrena para el problema concreto, eliminando la capa de clasificación final de la red y la volviendo a entrenarla con su conjunto de datos, ajustando los parámetros en todas las capas. Durante el entrenamiento, se cambia el tamaño de cada imagen a 299×299 píxeles para hacerla compatible con las dimensiones originales de la arquitectura de red Inception-v3 y aprovechar las características de imagen natural aprendidas por la red pre-entrenada de ImageNet. Este procedimiento, el *transfer learning*, es óptimo dada la cantidad de datos

disponibles. La CNN está entrenada utilizando propagación hacia atrás. Además, durante el entrenamiento, el número de imágenes se aumenta en un factor de 720 mediante *data augmentation*. Cada imagen se gira aleatoriamente entre 0 ° y 359 °. El mayor rectángulo vertical inscrito en el ojo se recorta de la imagen y se voltea verticalmente con una probabilidad de 0.5 [Esteva et al, 2017].

Para la detección de retinopatía en casos prematuros, también se emplea esta técnica. Para este problema se cuenta con una base de datos muy limitada, que además está formada por imágenes de baja resolución y las clases están bastante desbalanceadas, además de utilizar *transfer learning*, se emplea también *data augmentation*. La red pre-entrenada que se usa es GoogLeNet entrenada con el conjunto de datos de ImageNet, sustituyendo la última capa de esta por una capa densa con un clasificador *softmax* que clasifica en 2 clases. Típicamente se vuelve a entrenar en las imágenes médicas simplemente el clasificador lineal del final de la red, pero aquí se comprueba que se obtiene un mejor rendimiento si se incluyen algunas de las capas convolucionales dentro del 9º módulo Inception. Sin embargo, volver a entrenar demasiadas capas produce bastante sobreajuste, por lo que se usa un procedimiento iterativo para entrenar *n* capas finales, y si al comprar su rendimiento en validación con el de las *n - 1* capas anteriores este aumenta, se entrenan *n + 1* capas finales y así sucesivamente. Finalmente, se volvieron a entrenar las capas A, C, D, E y F del módulo Inception 9 que se observa en la figura 2-8 con el clasificador *softmax* de 2 nodos. Para el entrenamiento se emplean las imágenes de la base de datos recortadas con tamaño 240×240 píxeles.



(a) Inception module

Figura 2-6: Módulo Inception [Worral et al., 2016]

Por tanto, concluyen que entrenar el clasificador *softmax* desde capas anteriores de la red (no solo la última completamente conectada como ocurre en otros casos) no supone un problema, aunque se trate de capas más específicas de los conjuntos de datos, sin embargo, entrenar menos capas produce *underfitting*. Además, comprueban que añadir más capas totalmente conectadas al final de la red produce sobreajuste incluso utilizando *dropout*. Por otra parte, se observa que sobremuestrear la clase minoritaria también aporta buenos resultados teniendo un menor coste computacional que el *data augmentation* que produce un gran aumento en el número de imágenes en cada *epoch* [Worral et al., 2016].

En un último ejemplo, se desarrolla un sistema para cuantificar automáticamente la gravedad de la osteoartritis de rodilla. Para ello se cuenta con un dataset que contiene solo unos cuantos miles de imágenes (no llega a 10.000). Se utilizan dos enfoques distintos: en primer lugar, entrenan las redes VGG16, BVLC CaffeNet y VGG-M-128 con su conjunto de datos y extraen características de diferentes capas de las redes, como capas completamente conectadas, capas de *pooling* y convolucionales, para identificar el

conjunto de características más discriminatorio. Después entrenan SVMs lineales con las características extraídas de VGG para clasificar las imágenes de rodilla. En este caso, para clasificación binaria se observa que las características de conv4 layer y pool5 layer de VGG-M-128 net, y conv5 layer y pool5 layer de BVLC CaffeNet proporcionan una mayor precisión de clasificación en comparación con las capas fc6 y fc7 totalmente conectadas de las redes VGG y CaffeNet. Para la clasificación en varias clases ocurre algo similar, se obtiene mayor *accuracy* (precisión) entrenando los SVMs con las características de las capas convolucionales y de *pooling* que haciéndolo con las de las capas densas.

En segundo lugar, utilizan las redes BVLC CaffeNet, muy similar a AlexNet, y VGG-M-128. Ambas son más pequeñas y contienen menos capas y parámetros (∼62M) que VGG16 (∼138M). Reemplazan la última capa completamente conectada de ambas redes y reentrenan el modelo en su conjunto de datos usando *backpropagation*. Características de nivel inferior en las capas anteriores también se actualizan durante el *fine-tuning*. Se emplea una capa *softmax* estándar para la clasificación. En este enfoque, en el que se incluyen imágenes de rodilla volteadas a izquierda y derecha para aumentar el número de muestras de entrenamiento, el rendimiento de BVLC CaffeNet afinado fue ligeramente mejor que el de VGG-M-128. La disminución en las pérdidas y el aumento de precisión muestran que el *fine-tuning* es efectivo y hace que las características de la CNN sean más discriminatorias, lo que mejora la precisión de la clasificación. Las características extraídas de la capa densa proporcionan una clasificación ligeramente mejor en comparación con las capas de *pooling* y convolución [Antony et al., 2016].

Tras examinar estos trabajos, se comprueba que el *transfer learning* hace que se obtengan buenos resultados con bases de datos que no tienen gran tamaño, reduciendo además el tiempo de entrenamiento y el coste computacional de forma significativa, ya que se evita así el entrenamiento de millones de parámetros. En la mayoría de los casos usan Inception-v3 pre-entrenada con la base de datos de ImageNet con tamaños de entrada cercanos a los 299×299 píxeles que esta red tiene por defecto, aunque también emplean otras redes como VGG. Todos ellos realizan *fine-tuning*, aunque algunos eliminan solo la última capa de la red pre-entrenada y añaden un clasificador *softmax* entrenado en las imágenes de su base de datos obteniendo buenos resultados de *accuracy*, mientras que existe algún caso en el que entrenan también los pesos en distintas capas de la red porque obtienen mejores resultados en sus experimentos (aunque entrenar un número excesivo de capas les supone sobreajuste). Además, también lo combinan con otras técnicas como el *data augmentation.*

2.4.4.3 Combinaciones

La utilización de una única red, ya sea entrenada desde cero o mediante el uso de *transfer learning*, a menudo produce mucha variabilidad de resultados dependiendo de los valores iniciales de los pesos. Por ello, al combinar diversas predicciones, lo que se denomina *ensemble*, la robustez del sistema aumentará [Litjens et al., 2017]. A continuación, se describen los dos tipos mayoritarios de *ensembles* observados en la literatura: aquellos que combinan predicciones de distintas redes para la misma base de datos y los que combinan las predicciones de una misma red para diferentes versiones de las imágenes.

2.4.4.3.1 Ensembles de redes

Para la clasificación de DR en imágenes de fondo de ojo, se entrenan 3 redes distintas cuyas probabilidades de salida son usadas como entrada para un *random forest*:
La primera red consta de 11 capas convolucionales, seguidas de capas de *fractional max-pooling* (se trata de un *max-pooling* que en vez de tener un factor entero tiene uno fraccional, por ejemplo $\sqrt{2}$) y finalmente un clasificador *softmax* de 5 clases para predecir las clases 0, 1, 2, 3, 4. En las últimas 4 capas se utiliza aproximadamente el 10% de *dropout* ya que comprueban que aumentarlo más no supone una mejora. La segunda consiste en 14 capas, detalladas también en la imagen, y las técnicas utilizadas son las mismas que en la primera. En la tercera son 17 capas convolucionales seguidas de capas de *max-pooling* normal y clasificador *softmax* de 5 clases para predecir las clases 0, 1, 2, 3, 4. Las últimas 3 capas son completamente conectadas y en las dos primeras de ellas se utiliza el 50% de *dropout* [Graham, 2015].

En otro ejemplo utilizan un sistema compuesto por ocho redes neuronales convolucionales, todas ellas son una adaptación de la arquitectura VGGNet: un *ensemble* de dos redes para la clasificación de la gravedad de la DR, otros dos para la identificación de posible glaucoma y la identificación de AMD (age-related macular degeneration), y dos redes simples para evaluar la calidad de la imagen y para rechazar imágenes no válidas.

Para la clasificación de DR se utiliza un *ensemble* de dos redes. A una de ellas se le proporcionan las imágenes originales como entrada, mientras que a la otra se le proporcionan las imágenes con el contraste local normalizado. Las redes están formadas por varias capas convolucionales seguidas de capas *max-pooling* 2×2, con ReLU como función de activación y una capa densa con *dropout*. Finalmente, hay un clasificador *softmax* que contiene un nodo de salida para cada clase, indexados según la gravedad creciente de la clase de DR, de 0 a 4. El valor final es la media de las salidas de las dos redes convolucionales. La clasificación de las imágenes se realiza umbralizando esta puntuación final para el rendimiento de sensibilidad/especificidad deseado, estimado mediante el conjunto de validación. En este estudio se consideró adecuado un umbral de 0,70. El rendimiento obtenido es comparable y clínicamente aceptable para el modelo actual basado en la evaluación de imágenes de la retina por graduadores profesionales y además muestra consistencia en 10 conjuntos de datos de validación externos de múltiples etnias y entornos, utilizando diversos estándares de referencia en la evaluación de la retinopatía diabética por graduadores profesionales, optometristas, o especialistas en retina [Ting et al., 2017].

Por otra parte, también se ha observado que esta técnica se utiliza en otro tipo de problemas, como la segmentación. En un ejemplo de segmentación de membranas neuronales en imágenes de microscopio emplean 4 redes con arquitecturas ligeramente distintas que son entrenadas con las mismas imágenes previamente procesadas de distinta forma en algunos casos. Estas consisten en varias capas convolucionales, capas de *max-pooling*, y completamente conectadas, la última de estas con dos neuronas (1 por clase) que utilizan una función de activación *softmax* para garantizar que la activación de salida de

cada neurona pueda interpretarse como la probabilidad de que una imagen de entrada particular pertenezca a esa clase.

Las redes grandes con arquitecturas diferentes a menudo exhiben diferentes salidas a pesar de estar entrenadas en los mismos datos. Esto sugiere que estos clasificadores potentes y flexibles exhiben una variación relativamente grande pero un sesgo bajo. Por lo tanto, es razonable intentar reducir dicha variación promediando los resultados calibrados de varias redes con arquitecturas diferentes. Al promediar los resultados de las mismas se obtiene una puntuación significativamente mejor en todas las métricas que en las redes individuales [Ciresan et al., 2012].

Tras examinar distintos trabajos que utilizan la estrategia del *ensemble* de distintas redes, se ha observado que para la combinación de los resultados en algunos de ellos utilizan otras redes y en otros simplemente promedian las probabilidades. Este tipo de *ensembles* a menudo produce mejoras sustanciales en términos de precisión (*accuracy*), aunque conlleva un aumento en la complejidad y el coste computacional. Incluso agregar redes bastante débiles, es decir, redes cuyo rendimiento individual no es demasiado bueno, puede mejorar el rendimiento.

2.4.4.3.2 Ensemble de predicciones de la misma red

En este caso, se entrena una red varias veces con distintas versiones de las imágenes de la base de datos y se combinan los resultados aumentando así la robustez de las predicciones.

En primer lugar, en el trabajo anteriormente mencionado [Antony et al., 2015] extraen características (*best validation score, best kappa, final weights*) de la última capa *pooling* de las redes convolucionales que utilizan. Para aumentar la calidad de las características extraídas repiten la extracción hasta 50 veces (con diferentes *data augmentation* aleatorios como traslaciones, *stretching*, rotaciones, *flipping* y color) por imagen y utilizan la media y la desviación estándar de cada una de las características, estandarizadas para tener media cero y varianza unidad, como entrada para una 'red de mezcla' completamente conectada que consiste en varias capas densas. El score del *ensemble* sin la red de mezcla es de 0,824 mientras que tras la red de mezcla aumenta a 0.845.

Para el ejemplo anterior de clasificación de glaucoma en imágenes de fondo de ojo se entrena desde cero una red para lo cual utilizan *data augmentation* extrayendo regiones aleatorias y sus reflexiones horizontales de 224×224 de las imágenes originales de dimensiones 256×256. Para las imágenes de test, la CNN realiza una predicción extrayendo cinco regiones 224x224 de las 4 esquinas y el centro, así como sus reflexiones horizontales, y promediando las predicciones realizadas por la capa *softmax* de la red en estas 10 imágenes distintas extraídas de cada imagen original. Esto produce un aumento en el rendimiento de la red [Chen et al., 2015].

Por su parte, para clasificar lesiones de piel se utiliza una CNN compuesta de múltiples secciones donde cada sección considera la misma imagen con una resolución diferente. Una capa final combina las respuestas de varias resoluciones en una sola capa. Añaden una

capa de pérdida supervisada (es decir, una capa con una función de pérdida que compara la predicción con las etiquetas de cada imagen) a estas respuestas combinadas, lo que hace que la predicción final sea una función aprendida de múltiples resoluciones de la misma imagen. Esta pérdida se propaga hacia atrás en todas las secciones, lo que hace que toda la red se optimice con respecto a múltiples resoluciones de imagen. Al contar con un dataset limitado, utilizan la arquitectura pre-entrenada AlexNet (omitiendo la capa de salida específica de ImageNet para 1000 clases) para las primeras capas de la red, y capas adicionales sin entrenar que aprenden solo de imágenes de la piel para las capas de la red posteriores. Para pasar imágenes de diferentes resoluciones a través de todas las capas pre-entrenadas en una sola resolución, convierten (manteniendo los parámetros entrenados) capas completamente conectadas en capas convolucionales, ya que las capas convolucionales permiten entradas de tamaño variable.

Redimensionan las imágenes de la base de datos a 227×227 y 454×454 para la primera y segunda secciones de la red respectivamente. Realizan varios experimentos, entre ellos entrenar cada una de las secciones de la red por separado con imágenes de la misma resolución y entrenar la red con imágenes de distinta resolución y combinar las salidas como se ha explicado anteriormente. De ellos se extrae que el *accuracy* obtenido para el segundo experimento es mayor que para el primero. También realizan otro en el que además de combinar las probabilidades para diferentes resoluciones de la imagen utilizan distintos campos de visión, lo cual mejora aún más el *score* obtenido [Kawahara et al., 2016].

Por tanto, podemos concluir que al tener *datasets* limitados, combinar los resultados de diferentes versiones de la imagen (como copias rotadas, espejadas, con traslaciones, con diferentes resoluciones, o diferentes regiones de una imagen) para una red, mejora el score en todos los casos.

Por otra parte, en Gulshan et al, 2016, trabajo mencionado previamente, entrenan 10 veces la misma red con el mismo conjunto de datos, y la predicción final se calcula mediante un promedio lineal de las predicciones del *ensemble*. Utilizan la red neuronal pre-entrenada Inception-v3 para la detección de DR en imágenes de fondo de ojo. Esta se preinicializa utilizando pesos de la misma red entrenada para clasificar objetos en el conjunto de datos de ImageNet y después se entrena para las imágenes de la base de datos a clasificar. En este caso concreto ni siquiera se utilizan versiones distintas de las imágenes para cada entrenamiento y aun así se obtiene probabilidades distintas en cada uno de ellos.

El algoritmo de optimización utilizado para entrenar los pesos de la red es una implementación de descenso de gradiente estocástico distribuido, y para acelerar el entrenamiento se utiliza *batch normalization*. La red se entrena para realizar múltiples decisiones binarias, una para cada uno de los 4 niveles de retinopatía diabética, es decir, genera un número continuo entre 0 y 1 para cada una de las clasificaciones, que se corresponde con la probabilidad de que esa condición esté presente en la imagen. El rendimiento final es comparable con el de un oftalmólogo experto.

2.5 Deep Learning en medicina

En esta sección se van a comentar algunas de las principales aplicaciones del Deep Learning en imagen médica. El número de artículos sobre estas aplicaciones ha crecido mucho durante los últimos años, y los temas que más los ocupan son la segmentación de órganos y subestructuras, la detección, la clasificación de imágenes y exámenes, etc. de los que se hablará en los siguientes apartados.

2.5.1 Problemas de clasificación

Una de las primeras áreas donde se empezó a emplear Deep Learning fue la **clasificación de imágenes o exámenes**. La técnica estándar usada actualmente en este tipo de problemas son las Redes Neuronales Convolucionales (CNN). En la clasificación de exámenes, generalmente se tiene como entrada el examen, el cual consta de varias imágenes, y se quiere como salida una sola variable, por ejemplo, si la enfermedad está presente o no. Normalmente se tienen pocas imágenes disponibles por lo que la red no puede ser entrenada correctamente. La solución que se suele aplicar a este problema es el uso de distintas estrategias de *transfer learning*. En concreto, se suelen utilizar los dos tipos de estrategias explicadas anteriormente: [Shen et al., 2017]

- Usar una red pre-entrenada como extractor de características.
- Entrenar alguna capa de una red pre-entrenada en datos/imágenes médicas.

La segunda de ellas es utilizada en algunas publicaciones donde logran casi el rendimiento de un humano experto. Por ejemplo, la CNN Inception-v3 de GoogleNet pre-entrenada con 1.28 millones de imágenes para el *ImageNet Large Scale Visual Recognition Challenge* de 2014 es entrenada para el problema concreto de clasificación de lesiones de la piel, usando 757 clases de enfermedades.

El *dataset* usado en este artículo está compuesto por imágenes etiquetadas por 21 dermatólogos organizadas en una taxonomía estructurada de árbol de 2.032 enfermedades, y además se divide en 127.463 imágenes de entrenamiento y validación y 1.942 imágenes de test etiquetadas mediante el resultado de una biopsia. Previamente, desarrollan un pequeño algoritmo que divide las enfermedades en clases de entrenamiento según esta taxonomía.

Para la validación, usando solo imágenes comprobadas por biopsia, se prueba si el algoritmo y los dermatólogos podrían distinguir lesiones malignas de benignas de origen epidérmico (carcinomas de queratinocitos o queratosis seborreicas benignas) o melanocítico (melanoma maligno o nevo benigno), lo cual es muy complicado ya que lesiones malignas y benignas comparten muchas características visuales.

Las métricas de comparación utilizadas son sensibilidad y especificidad, donde $\text{sensibilidad} = \frac{verdadero\ positivo}{positivo}$ y, $\text{especificidad} = \frac{verdadero\ negativo}{negativo}$.

En el resultado obtenido de esta validación se puede observar que el rendimiento de la CNN supera a cualquier dermatólogo cuya sensibilidad y punto de especificidad se

encuentren por debajo de la curva azul, que representa la CNN, y la mayoría están por debajo.

Este artículo es un buen ejemplo del gran potencial del Deep Learning en imagen médica, ya que, usando una sola red neuronal convolucional pre-entrenada y entrenándola en la clasificación de lesiones cutáneas, esta es capaz de igualar o incluso superar el rendimiento de al menos 21 dermatólogos veteranos. Además, piensan que al ser un método rápido y escalable podría ser implementado en dispositivos móviles [Esteva et al, 2017].

Esta misma estrategia se utiliza para clasificar imágenes de retinopatía diabética, problema en el que se centra este trabajo. En este problema, para el desarrollo del algoritmo se utilizan imágenes de fondo de retina obtenidas de la base de datos EyePACS en Estados Unidos y 3 hospitales en India, y para evaluarlo se hace mediante dos conjuntos de validación, uno que consiste en una muestra aleatoria de imágenes tomadas en los sitios de detección EyePAC, y el segundo es el *dataset* Messidor-2 que está disponible públicamente. Las imágenes fueron calificadas por oftalmólogos según la presencia de retinopatía diabética, edema macular diabético y calidad de imagen. Además, también calificaron la gravedad de la retinopatía diabética (ninguna, leve, moderada, grave o proliferativa) de acuerdo con la escala de Retinopatía clínica internacional.

La red neuronal utilizada es Inception-v3 preinicializada utilizando los pesos de la red entrenada para el conjunto de datos de ImageNet. Se trata de una única red entrenada para realizar múltiples decisiones binarias, como retinopatía diabética moderada o peor, retinopatía diabética grave o peor, o edema macular. La red neuronal entrenada genera un número continuo entre 0 y 1 para cada una de las clasificaciones, que se corresponde con la probabilidad de que esa condición esté presente en la imagen.

El rendimiento del algoritmo se mide mediante el área bajo la curva ROC (Receiver Operating Curve) que representa sensibilidad frente a 1 - especificidad. El área bajo la curva ROC obtenida para el conjunto de validación de la base de datos EyePACS-1 es de 0.991 y para el de la base de datos Messidor-2 de 0.990 [Gulshan et al, 2016].

Mediante estos resultados se demuestra que las redes neuronales profundas pueden ser entrenadas usando grandes bases de datos sin ser necesario especificar características concretas de las lesiones. Además, cuentan con una gran sensibilidad y especificidad, y con una ventaja que las evaluaciones humanas no poseen, que es la consistencia en sus interpretaciones, es decir, que siempre van a predecir lo mismo. Este problema de detección de retinopatía diabética ya ha sido estudiado por otros grupos previamente. Recientemente en la competición de Kaggle *'Diabetic Retinopathy Detection'* se utilizó Deep Learning para predecir la presencia de retinopatía diabética (sin contar con el edema macular). En ella, el modelo ganador logró un rendimiento que puede ser comparable con el de un oftalmólogo experto.

Tras el estudio de la literatura en el problema de clasificación que es el que se va a centrar este trabajo, se han encontrado gran cantidad de técnicas y variantes a la hora de abordarlo, por lo que el objetivo principal del trabajo se centrará en contrastar distintas estrategias en la misma base de datos para poder determinar sus ventajas e inconvenientes.

Por otra parte, también se utiliza Deep Learning para **clasificar objetos o lesiones** presentes en una zona determinada. Se utilizan pequeñas partes de imágenes (previamente

recortadas), que se clasifican en dos o más clases. Por ejemplo, se podría clasificar nódulos presentes en un pulmón, en benignos o malignos. En muchos problemas de este tipo se requiere combinar tanto la información sobre la apariencia de la lesión como la información contextual global sobre la ubicación de la lesión para poder conseguir una clasificación precisa, lo cual las arquitecturas de Deep Learning comunes no permiten llevar a cabo. En general, en clasificación de objetos o lesiones se usan también CNNs, pero no suelen utilizarse redes pre-entrenadas tanto como en clasificación de exámenes, aunque sí que incorporan algún tipo de estrategia distinta como las arquitecturas *multi-stream* debido a la necesidad de combinar la información contextual antes mencionada, que no se van a entrar a examinar [Litjens et al., 2017] [Shen et al., 2015b].

2.5.2 Problemas de detección

En primer lugar, la **detección de objetos de interés o lesiones** es uno de los pasos clave en el diagnóstico o el seguimiento de enfermedades ya que permite un análisis cuantitativo de parámetros como volumen y forma, y además se trata de una tarea que consume gran cantidad de tiempo a los profesionales. Debido a esto, es uno de los problemas más investigados desde siempre en el campo del tratamiento automático de imagen médica. La principal diferencia con clasificación de objetos es que en este tipo de problemas se debe clasificar cada píxel de la imagen, por ello, se usan arquitecturas muy similares a las de clasificación de objetos en las que cada píxel de la imagen donde se quiere detectar la lesión es un objeto a clasificar, con algunas diferencias evidentemente. Por tanto, en la mayoría de los artículos referidos a este tipo de problema se decantan por el uso de CNNs *multi-stream*.

Además, también se emplea Deep Learning para la **localización automática de órganos o regiones**, paso muy importante en labores de segmentación, pero también en planificación de terapias o intervenciones quirúrgicas [Litjens et al., 2017].

2.5.3 Problemas de segmentación

La **segmentación de órganos y subestructuras** es el tema más común en los artículos sobre aprendizaje profundo en imágenes médicas, y también donde se encuentra la variedad más amplia en metodología. En los artículos más recientes se puede ver cómo destaca el uso de arquitecturas basadas únicamente en CNNs, como por ejemplo U-net [Ronneberger et al., 2015]. No obstante, las Redes Recurrentes Profundas (DRNNs), se han vuelto más populares recientemente para las tareas de segmentación. Estas son un tipo de redes inicialmente propuestas para aplicaciones con entradas unidimensionales, como predicción de series temporales, reconocimiento de voz, modelado de lenguaje, traducción, subtítulos de imágenes, etc. pero que cada vez se aplican más a imágenes. Su uso en problemas de segmentación se ha hecho más popular en los últimos años y se pueden encontrar varios ejemplos de ello, pero no se entrará a analizarlos en profundidad en este trabajo.

Por otra parte, se puede encontrar otro uso del Deep Learning en **segmentación de lesiones**. La segmentación de lesiones combina los desafíos de la detección de objetos y la segmentación de órganos y subestructuras, por lo que los algoritmos que se emplean actualmente serán similares a los de estas dos tareas [Litjens et al., 2017].

2.5.4 Otros problemas

Además de las ya mencionadas, se pueden encontrar multitud de aplicaciones de Deep Learning en imagen médica, como, por ejemplo:

El **registro**, es decir, la alineación espacial de distintas imágenes médicas, que consisten en calcular una transformación de coordenadas de una imagen a otra, para compararlas, integrar la información, seguimientos de evolución etc. A diferencia de la clasificación y la segmentación, los artículos sobre el tema son algo escasos y con enfoques muy diferentes.

Por otro lado, la **recuperación de imágenes basada en contenido** (CBIR) es una técnica para identificar historias de casos similares o relacionados en bases de datos masivas. El desafío al que se enfrentan los métodos CBIR es aprender características efectivas en múltiples niveles de abstracción a partir de información a nivel de píxel, en el cual las CNN tienen una gran capacidad, por lo que la mayoría de los enfoques actuales usan CNNs pre-entrenadas para esta tarea.

Además, existen muchas otras aplicaciones como pueden ser la generación de informes de texto a partir de imágenes, sistemas que además de la imagen admiten una descripción en texto para mejorar su predicción, la mejora de imágenes y/o eliminación de 'obstáculos', como por ejemplo de huesos en radiografías, etc [Litjens et al., 2017].

2.6 Herramientas (software)

Además de las distintas arquitecturas, técnicas para controlar el sobreajuste, etc., se han estudiado algunas características de los distintos *software* como librerías o lenguajes más populares para *deep learning* a fin de seleccionar los más adecuados para emplear en este trabajo.

A medida que la popularidad del aprendizaje profundo se ha incrementado en los últimos años, han aparecido varios *frameworks* de software de *deep learning* para permitir el desarrollo e implementación eficientes de estos métodos. Aunque la lista es muy extensa se han estudiado algunos de los más populares, como es el caso de Microsoft Cognitive Toolkit (CNTK), Theano, MXNet, Torch y Pytorch. En primer lugar, Microsoft Cognitive Toolkit, está disponible para Windows, Linux y macOS (via Docker on roadmap), y tiene varias interfaces como Python, C++, Command line o BrainScript [Microsoft Cognitive Toolkit]. Por su parte Theano, cuya única interfaz es Python, es un *cross-platform sotware*, es decir, que se puede implementar en varias plataformas. Ambas pueden ser utilizadas como backend de Keras. MXNet es escalable, permite un rápido entrenamiento de los modelos y es compatible con un modelo de programación flexible y múltiples lenguajes de programación (incluidos C ++, Python, Julia, Matlab, JavaScript, Go, R, Scala, Perl y Wolfram Language), y además está disponible en Linux, macOS, Windows, AWS, Android, iOS y JavaScript [Theano]. Torch, que ya no está en desarrollo activo, está disponible para Linux, macOS, Windows, Android e iOS y sus interfaces serían Lua, LuaJIT, C, además de ser una librería para C++/OpenCL [Collobert et al., 2011]. Finalmente, PyTorch, basada en Torch, que se utiliza para aplicaciones como el procesamiento de lenguaje natural, está para Linux, macOS y Windows, y su interfaz es

Python. Todos ellos tienen en común que son de código abierto, disponen de redes pre-entrenadas y permiten tanto CPU como GPU [Bahrampour, 2015].

Por otro lado, la librería más destacada para *deep learning* es Tensorflow. Se trata de una librería de software de código abierto para la programación de flujo de datos en una variedad de tareas. Es una librería de matemáticas simbólica, y también se usa para aplicaciones de aprendizaje automático como las redes neuronales. TensorFlow fue desarrollado por el equipo de Google Brain para uso interno de Google. Se puede utilizar tanto en Linux, macOS y Windows como en Android. Es compatible con mútiples lenguajes de programación como Python, C/C++, Java, Go, JavaScript, R, Julia y Swift [Abadi et al., 2015].

Keras es una librería de código abierto para redes neuronales escrita en Python [Keras]. Se ejecuta sobre TensorFlow, Microsoft Cognitive Toolkit o Theano. Está diseñada para ser fácil de usar, modular y extensible. Ofrece un conjunto de abstracciones más intuitivo y de mayor nivel que facilita el desarrollo de modelos de aprendizaje profundo, independientemente del backend computacional utilizado. Microsoft también agregó un backend CNTK a Keras. Contiene numerosas implementaciones de bloques de construcción de redes neuronales de uso común, como capas, objetivos, funciones de activación, optimizadores y una gran cantidad de herramientas para facilitar el trabajo con datos como imágenes y texto. Además de las redes neuronales estándar, tiene soporte para redes neuronales convolucionales y recurrentes. Admite otras capas de utilidad comunes, como el *dropout*, *batch normalization* y *pooling*. Permite a los usuarios producir modelos profundos en smartphones (iOS y Android), en la web o en la máquina virtual de Java. Disponen de gran cantidad de modelos pre-entrenados y permiten el uso de entrenamiento distribuido de modelos de aprendizaje profundo en clusters de GPUs y TPUs [Keras].

Finalmente, los principales lenguajes de programación son Python, R y MATLAB. MATLAB, es un entorno de computación numérica multi-paradigma y lenguaje de programación patentado desarrollado por MathWorks. Permite la manipulación de matrices, el trazado de funciones y datos, la implementación de algoritmos, la creación de interfaces de usuario y la interacción con programas escritos en otros lenguajes, incluidos C, C++, C#, Java, Fortran y Python. Deep Learning Toolbox™ de MATLAB (anteriormente Neural Network Toolbox™) proporciona un marco de pruebas para diseñar e implementar redes neuronales profundas con algoritmos, modelos previamente entrenados y apps. Puede utilizar redes neuronales convolucionales (ConvNets y CNNs) y redes Long-Sort Term Memory (LSTM) para realizar clasificación y regresión en imágenes, series temporales y datos de texto. No es de código abierto. Se puede utilizar en Windows, Linux y macOS. Dispone de modelos pre-entrenados (incluidos SqueezeNet, Inception-v3, ResNet-101, GoogLeNet y VGG-19) y modelos importados desde TensorFlow™-Keras y Caffe. Se trata de un lenguaje muy estable, pero al ser más reciente cuenta con un menos desarrollo que Python y R. Por su parte tanto Python como R son *opensource* y permiten el uso de Tensorflow, pero tienen más problemas de estabilidad que MATLAB y no son compatibles hacia atrás [Deep Learning Toolbox].

3 Metodología

Este capítulo aborda la descripción de las diferentes etapas que conforman el diseño del sistema de clasificación. Se expone un análisis de la base de datos empleada y su preprocesado, la descripción de las distintas arquitecturas seleccionadas y finalmente el software utilizado para todo ello.

3.1 Dataset

El *dataset* empleado en este trabajo es la base de datos EyePACS de imágenes de fondo de retina para la detección de retinopatía diabética disponible por completo públicamente. Dicha base de datos ha sido descargada en Kaggle, donde se proporciona para la competición *'Diabetic Retinopathy Detection'*.

La retinopatía diabética es la principal causa de ceguera en la población de 20 a 64 años del mundo desarrollado. La Organización Mundial de la Salud estima que 347 millones de personas tienen la enfermedad en todo el mundo. La retinopatía diabética (DR) es una enfermedad ocular asociada con la diabetes de larga duración, que afecta hasta al 80% de las personas que han tenido diabetes durante 20 años o más. La progresión del deterioro de la visión puede reducirse o evitarse si se detecta DR a tiempo, sin embargo, esto puede ser difícil ya que la enfermedad a menudo muestra pocos síntomas hasta que es demasiado tarde para proporcionar un tratamiento efectivo.

Actualmente, la detección de DR es un proceso que requiere que un médico capacitado examine y evalúe las fotografías digitales de fondo de retina. Los médicos pueden identificar la DR por la presencia de lesiones asociadas con las anormalidades vasculares causadas por la enfermedad, pero la experiencia y el equipo requeridos a menudo faltan en áreas donde la tasa de diabetes es alta y la detección de DR es más necesaria. A medida que la cantidad de personas con diabetes continúa creciendo, la infraestructura necesaria para prevenir la ceguera debida a la DR será aún más insuficiente. Hace tiempo que se reconoce la necesidad de un método integral y automatizado para la monitorización de DR, y los esfuerzos previos han progresado satisfactoriamente utilizando la clasificación de imágenes, el reconocimiento de patrones y el aprendizaje automático. El objetivo de la competición de Kaggle es el desarrollo de un sistema de detección automatizado, con las fotografías de fondo de retina a color como entrada, buscando modelos con potencial clínico realista. En ella, el modelo ganador logró un rendimiento que puede ser comparable con el de un oftalmólogo experto [Diabetic Retinopathy Detection].

El *dataset* consta de un total de 88.702 imágenes de fondo de retina de alta resolución tomadas en variedad de condiciones. De estas, 35.126 van a formar parte del conjunto de entrenamiento y 53.576 del conjunto de test. Están etiquetadas con una identificación de sujeto, y con la palabra '*left*' si se trata del ojo izquierdo y '*right*' si se trata del derecho (por ejemplo, 1_left.jpeg es el ojo izquierdo del paciente identificado como 1).

Las imágenes del conjunto de datos provienen de diferentes tipos de cámaras, por lo que van a tener distintas formas, tamaños y resoluciones, las cuales van a ir desde las pocas centenas de KBytes hasta unos pocos MBytes, y pueden ser diferentes incluso entre las

imágenes del ojo izquierdo y derecho del mismo paciente. Al igual que cualquier otro conjunto de datos del mundo real, estas imágenes pueden estar desenfocadas, subexpuestas o sobreexpuestas, contener objetos extraños, etc. Algunas imágenes muestran cómo se vería la retina anatómicamente (mácula a la izquierda, nervio óptico a la derecha para el ojo derecho), y otras cómo se vería a través de una lente de microscopio, es decir, invertida. La imagen está invertida si la mácula está situada ligeramente más arriba que el nervio óptico, es decir, si la mácula está por debajo del nervio óptico no está invertida. Además, si la imagen no está invertida, va a presentar una pequeña muesca en el lateral del ojo con forma de cuadrado, círculo o triángulo.

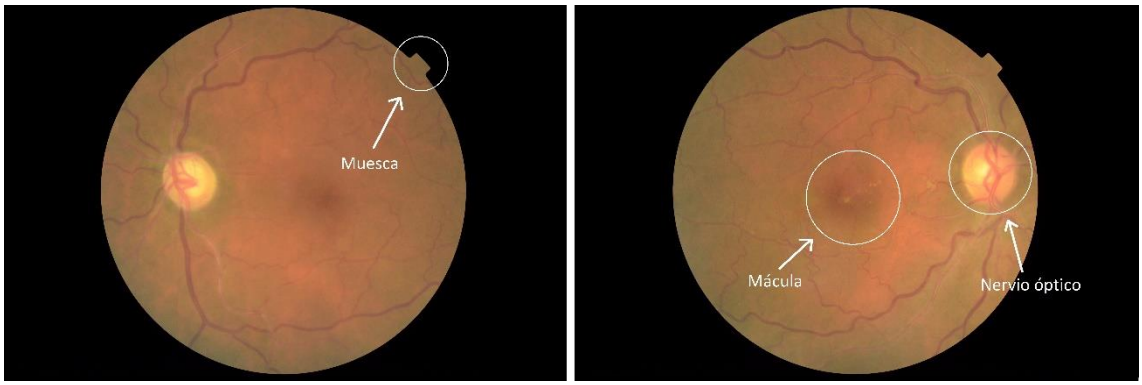


Figura 3-1: Imágenes no invertidas. Ojos izquierdo y derecho respectivamente

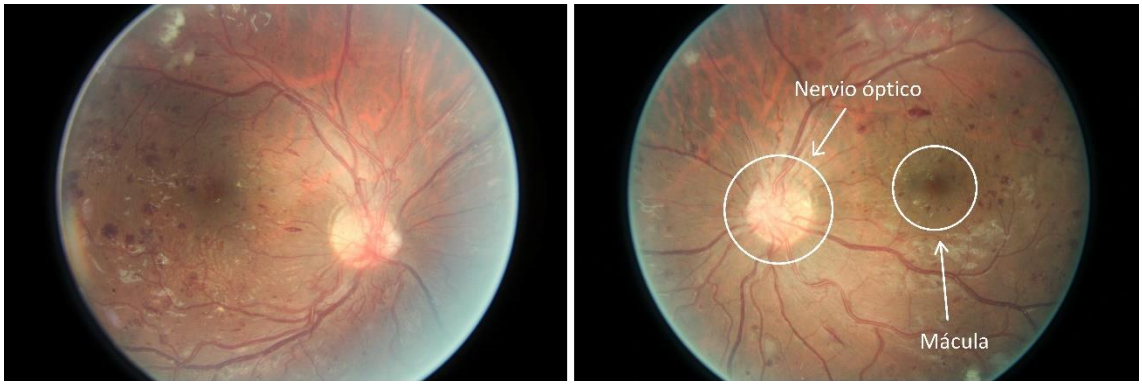


Figura 3-2: Imágenes invertidas. Ojos izquierdo y derecho respectivamente

La presencia de retinopatía diabética en cada una de las imágenes ha sido evaluada por médicos expertos de acuerdo con la siguiente escala:

- 0 (Sin DR)
- 1 (DR Leve)
- 2 (DR Moderada)
- 3 (DR Grave)
- 4 (DR proliferativa)

Figura 3-3: Ejemplos imágenes etiquetadas del 0 al 4 respectivamente

Con la base de datos se adjunta un documento Excel donde se etiquetan cada una de las imágenes de entrenamiento según esta escala. Como se puede observar en la siguiente tabla, en la primera columna se detalla el nombre del archivo según la identificación de sujeto y si se trata de una imagen de ojo izquierdo o derecho como se ha comentado anteriormente, mientras que en la segunda se encuentra el nivel (de 0 a 4) que le corresponde a cada imagen según la graduación anterior.

image	level
10_left	0
10_right	0
13_left	0
13_right	0
15_left	1
15_right	2
16_left	4
16_right	4

Tabla 3-1: Extracto del documento Excel con las etiquetas del conjunto de entrenamiento

En esta fase del trabajo, se analizó la base de datos extrayendo una serie de conclusiones adicionales a la información anterior, las cuales forzaron un ajuste del *dataset*. Dichas características sobre la base de datos son las siguientes:

- Las etiquetas del conjunto de test no son proporcionadas junto con la base de datos, por ello solo se hace uso de aquellas imágenes pertenecientes al conjunto de entrenamiento, que posteriormente se dividirá en los conjuntos de entrenamiento, validación y test. A partir de ahora se trabaja solo sobre este conjunto de los datos formado por 35.126 imágenes.

- Existen algunos archivos corruptos, en concreto 5, en los que la imagen aparece cortada, es decir, se han perdido algunos píxeles. Estas imágenes han sido eliminadas del *dataset* junto con la imagen del otro ojo correspondiente al mismo paciente.

- Además, al realizar la parte del preprocesamiento de las imágenes, explicado a continuación, se han detectado varias de ellas cuyo contraste era tan bajo que dificultaba mucho la diferenciación de la parte del ojo del fondo negro. También se

ha optado por prescindir de estas imágenes y de las correspondientes al otro ojo del mismo sujeto, lo que supone un total de 26 imágenes.

Por tanto, finalmente la base de datos utilizada al completo está formada por 35.100 imágenes, de las cuales un 55% (19.305) van a formar parte del conjunto de entrenamiento, un 15% (5.265) del conjunto de validación y un 30% (10.530) del conjunto de test.

3.2 Tipos de preprocesamiento de las imágenes

Como se ha explicado anteriormente las imágenes tienen tamaños diferentes y poseen una gran zona negra alrededor que lógicamente no aporta nada a la extracción de características de estas, ya que en esencia es ruido. Además, algunas de ellas pueden estas sobreexpuestas, o tener un bajo contraste, etc. Por todo ello, y siguiendo lo hallado en el estudio del estado del arte, se estudian los siguientes preprocesamientos a las imágenes.

3.2.1 Recorte y redimensionado

- **Recortar el fondo negro y redimensionar.** En primer lugar, se recorta todo el fondo negro posible calculando el rectángulo delimitador de las regiones distintas de cero en la imagen como se muestra en la siguiente figura.

Figura 3-4: Imágenes recortadas

Posteriormente se redimensiona esta nueva imagen a cuadrados de la resolución deseada, por lo que la imagen se deformará más cuanto más desiguales sean estas dimensiones. Esto ocurre sobre todo en aquellas imágenes que están cortadas y no aparece el círculo del ojo entero, como en la del segundo ejemplo, ya que al recortarlas queda un rectángulo de lados bastante desiguales [Antony et al., 2015].

Figura 3-5: Imágenes recortadas y redimensionadas

- **Recortar el fondo negro, rellenar con ceros el lado más corto del rectángulo para tener un cuadrado, y redimensionar.** Primero se recorta el fondo negro del mismo modo que se hace anteriormente. Después, en caso de que ancho y alto no sean iguales (como ocurre en el segundo ejemplo) se rellena con ceros (negro) la dimensión menor para igualarla a la mayor y obtener así una imagen cuadrada, de forma que al redimensionarla esta no se deforme, y después redimensiona esta nueva imagen a la resolución deseada [Berger, 2015].

Figura 3-6: Imágenes recortadas, rellenadas con ceros y redimensionadas

- **Recortar cuadrado de la imagen fijando centro y dimensiones de este y redimensionar.** En este caso se recortan todas las imágenes en cuadrados centrados en el centro de la imagen y de las dimensiones deseadas, por lo que recorta el fondo negro y también parte del ojo. Como hay imágenes con dimensiones muy distintas (desde más de 2000 píxeles hasta menos de 300), se comprueba para cada imagen si alguna de sus dimensiones es menor que las dimensiones del cuadrado a recortar, y si es así se reducen las dimensiones del cuadrado. Se parte de las dimensiones 1800x1800 y se llega hasta algunos con dimensiones 200x200. Posteriormente redimensiona la imagen a la resolución deseada [Chase, 2017].

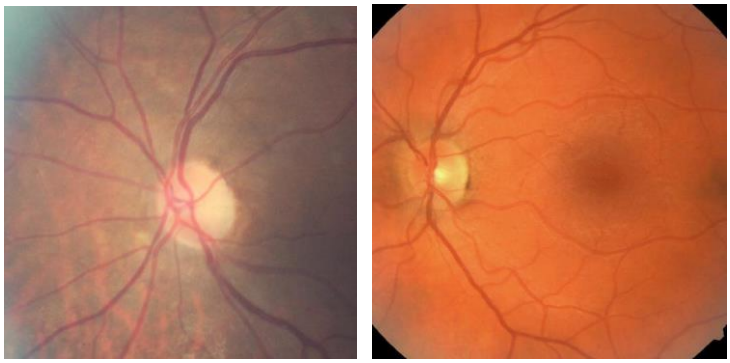


Figura 3-7: Imágenes recortadas

3.2.2 Estandarización del contraste/iluminación, mejora de la calidad de la imagen

- **Ecualización de histograma.** Consiste en una transformación del histograma de la imagen de forma que este tenga una distribución uniforme para mejorar su contraste. Se realiza transformando la imagen al espacio LAB y aplicando la ecualización solo sobre el canal L (luminosidad). Para ello solo se tienen en cuenta el valor de los píxeles correspondientes al ojo, no aquellos que forman parte del fondo negro [Berger, 2015].

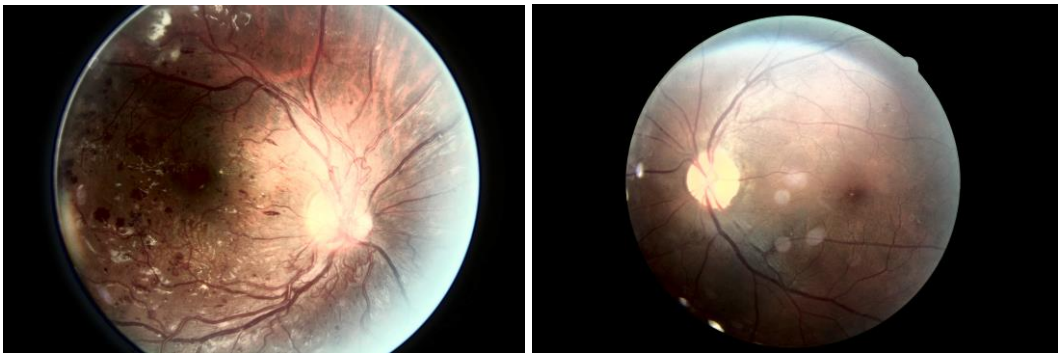


Figura 3-8: Imágenes con ecualización de contraste

- **CLAHE.** Se realiza la ecualización del histograma, pero no considerando el contraste global de la imagen, sino que se trata de una ecualización adaptativa. También se lleva a cabo transformando la imagen al espacio LAB y aplicando la ecualización solo sobre el canal L y sin tener en cuenta el fondo negro [Rasta et al., 2017] [Histogram Equalization].

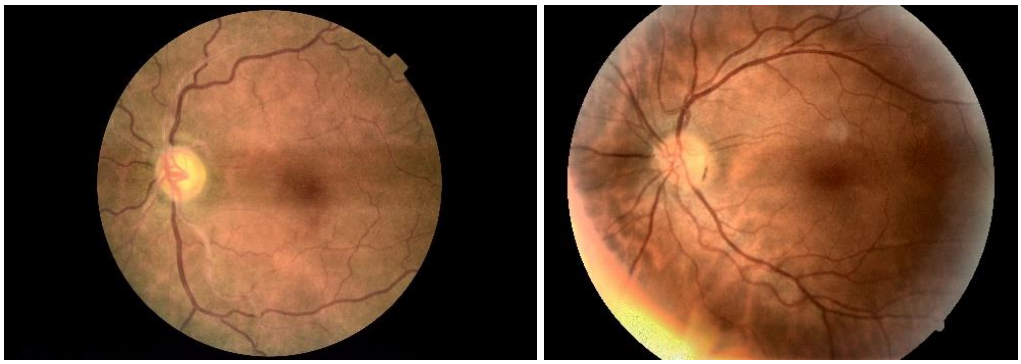


Figura 3-9: Imágenes con ecualización CLAHE

- **Normalización de color.** La base de datos contiene imágenes con niveles de iluminación extremadamente variados, lo cual afecta a los valores de intensidad de píxeles y crea una variación innecesaria que no está relacionada con los niveles de clasificación. Para contrarrestar esto, en algunos de los trabajos estudiados se implementa la normalización del color como la que se puede observar en las siguientes imágenes [Pratt et al., 2016] [Graham, 2015].

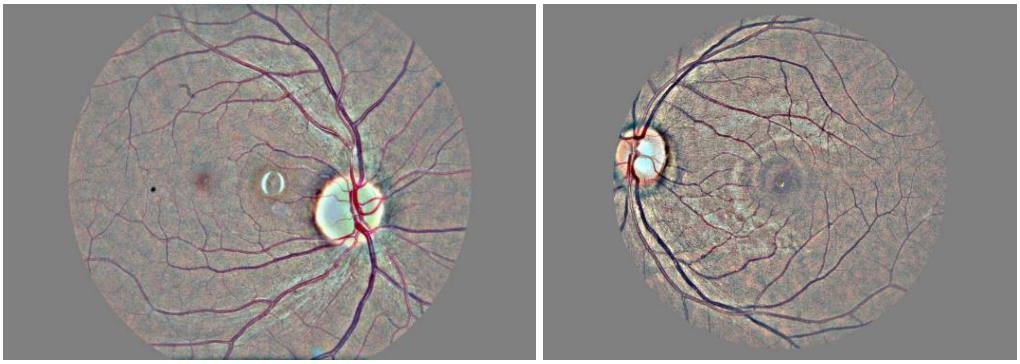
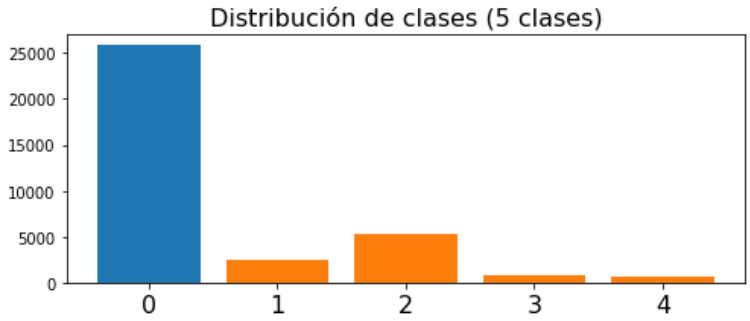


Figura 3-10: Imágenes con color normalizado

3.3 Balanceo de clases

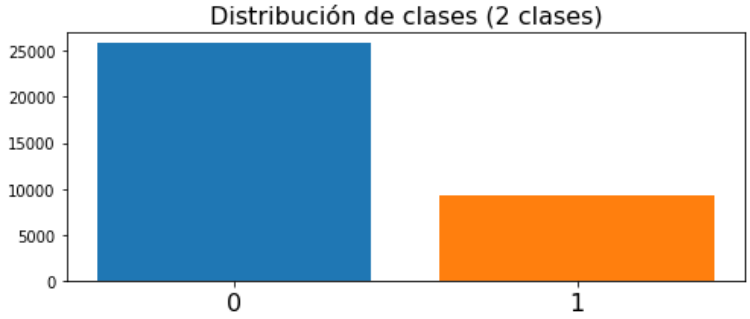
Al analizar las etiquetas del conjunto de entrenamiento se advirtió que, si bien hay cinco categorías diferentes a las que pertenece cada imagen, existe un gran desequilibrio entre las distintas clases en el conjunto de datos original. La distribución de las clases se muestra en la siguiente gráfica:



Clase	nº	%
0	25796	73.49
1	2435	6.94
2	5288	15.06
3	873	2.49
4	708	2.02
Total	35100	

Figura 3-11: Distribución de clases (5 clases)

Incluso tomando las clases minoritarias 1, 2, 3 y 4 como una única clase nueva, identificada con la etiqueta 1 (con DR), las dos clases resultantes siguen estando fuertemente desbalanceadas:

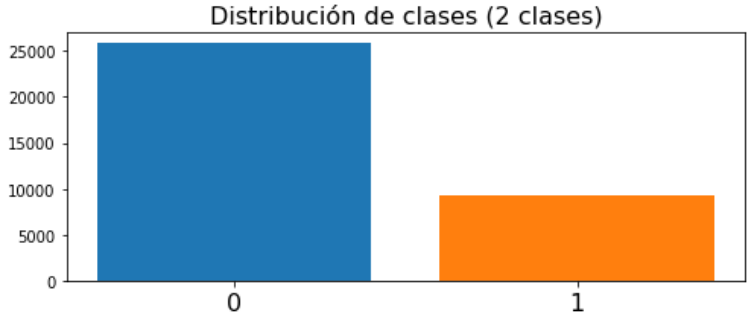


Clase	nº	%
0	25796	73.49
1	9304	26.51
Total	35100	

Figura 3-12: Distribución de clases (2 clases)

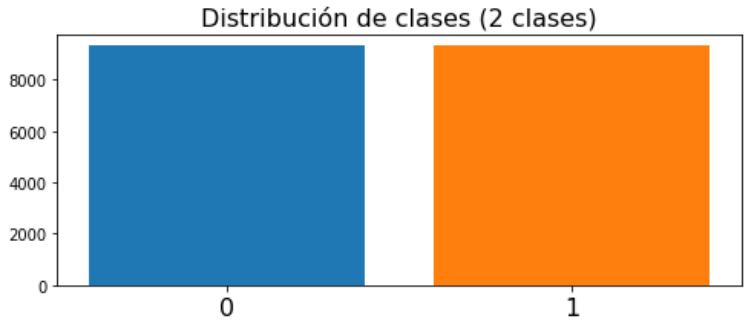
A fin de rectificar este desequilibrio, se han llevado a cabo una serie de técnicas que se explican a continuación.

- ***Undersampling*.** Se reduce la clase mayoritaria, es decir, la clase etiquetada como 0. Para ello se eliminan imágenes pertenecientes a dicha clase de forma aleatoria hasta quedar igualada con la clase minoritaria en la división en dos únicas clases, resultando la siguiente distribución:



Clase	nº	%
0	25796	73.49
1	9304	26.51
Total	35100	

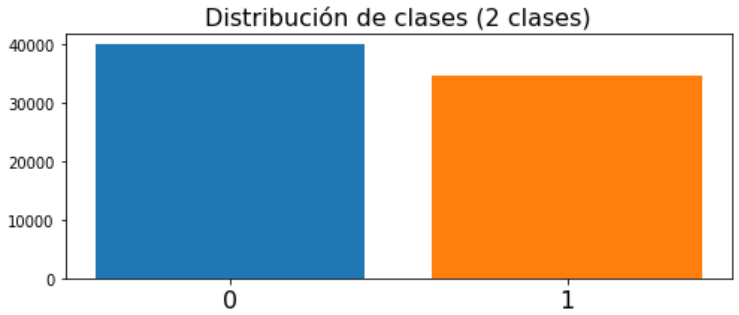
Figura 3-13: Distribución de clases (2 clases)



Clase	nº	%
0	9304	50
1	9304	50
Total	18608	

Figura 3-14: Distribución de clases balanceadas (2 clases)

- ***Oversampling*.** Se realiza *data augmentation*. Basándonos en el estudio del estado del arte, se realizan copias de las imágenes reflejadas tanto vertical como horizontalmente, y rotadas 90, 120, 180 y 270º. En concreto, las imágenes de la clase mayoritaria solo se reflejan sobre su eje horizontal, creando una única copia de cada una de ellas mientras que de las pertenecientes a las clases minoritarias se realizan 5 copias, una reflejada sobre su eje vertical y 4 rotadas, a fin de balancear las clases de la misma forma que anteriormente. La distribución obtenida es la siguiente:



Clase	nº	%
0	39996	53.45
1	34829	46.55
Total	74825	

Figura 3-15: Distribución de clases *data augmentation* (2 clases)

3.4 Arquitecturas de las redes que se estudiarán

Tras el estudio de la literatura existente, se han hallado diversas arquitecturas mediante las que afrontar este tipo de problemas de imagen médica. Podemos agrupar los trabajos consultados, en general, en dos grandes grupos: aquellos que emplean arquitecturas entrenadas desde cero en el problema en cuestión, y aquellos que utilizan *transfer learning*. Tras observar que en aquellos trabajos que entrenan redes desde cero utilizan recursos de los que no disponemos (como GPUs, etc.) para el ajuste de los millones de parámetros con los que cuentan estas redes tan profundas, se ha optado por centrarse en la técnica del *transfer learning*, es decir, en la utilización de redes pre-entrenadas.

En concreto, y basado en el estudio del estado del arte se van a emplear las siguientes redes pre-entrenadas con la base de datos ImageNet, ya que son las más empleadas en los trabajos relacionados con este tema:

- **Inception-v3.** Esta red tiene una profundidad de 159 capas, de las cuales la mayoría son capas convolucionales, aunque también hay varias de *pooling* (tanto *max-pooling* como *average pooling*), de *dropout* y densas. Además, tiene 23.851.784 parámetros y un accuracy top-1 de 0.779 y top-5 de 0.937. Su arquitectura es la siguiente:

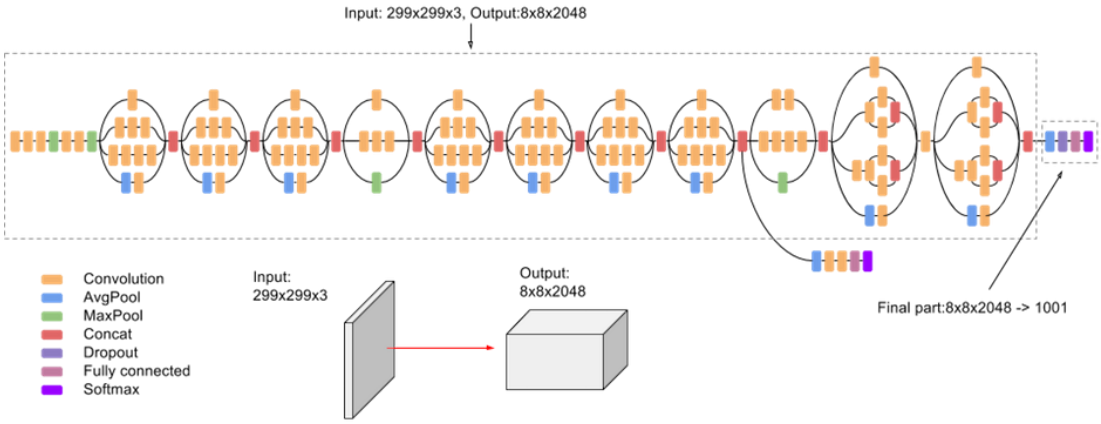


Figura 3-16: Arquitectura de la red Inception-v3 [fuente]

- **VGG16.** Se trata de una red bastante menos profunda, de 23 capas, aunque el número de parámetros que se entrenan es mucho mayor (138,357,544). Su accuracy top-1 es 0.713 y el top-5 0.901. Su arquitectura, también formada por capas convolucionales, de *pooling* y densas, es la siguiente:

Figura 3-17: Arquitectura de la red VGG16 [fuente]

De la primera de ellas se obtendrán, en caso de que el tamaño de la imagen de entrada sea 299x299 píxeles (el tamaño por defecto), 8x8x2048 valores de salida por cada imagen, que tras pasar por una capa de *average pooling* (en la que estos valores se promedian), resultan 2048 valores por imagen. De la segunda, VGG16 cuyo tamaño de entrada por defecto es 224x224, se obtienen 7x7x512 valores de salida que tras la capa de *average pooling* serán 512. Estos valores son pasados como entrada a una nueva red clasificadora formada por capas densas y de *dropout*, cuya última capa se encarga de dar como salida la probabilidad de que la imagen pertenezca a cada clase, obteniendo así nuestra clasificación. Tiene la siguiente arquitectura:

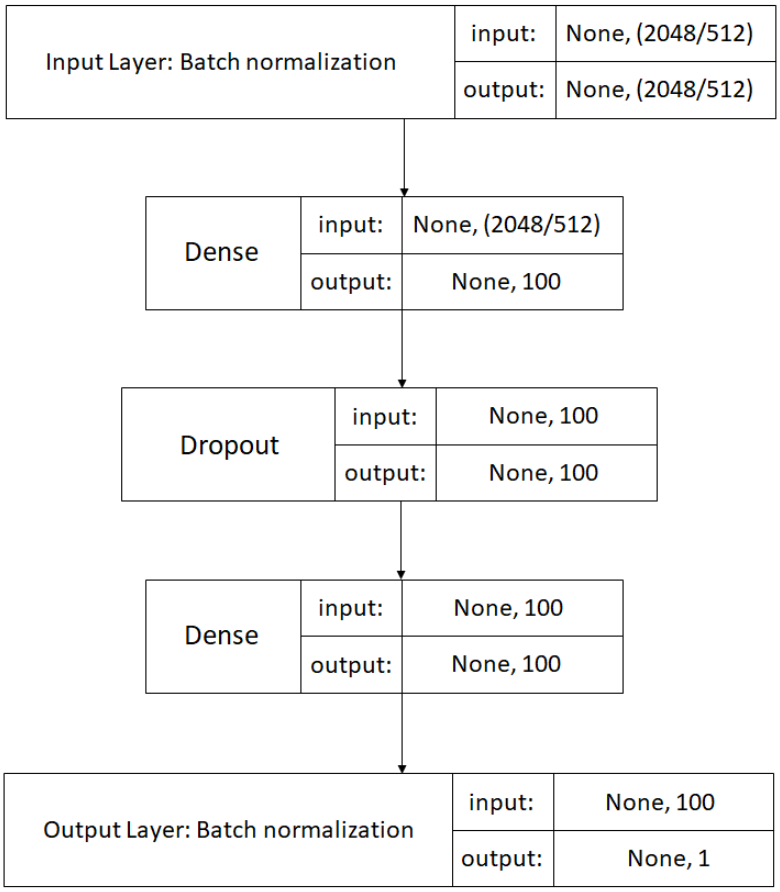


Figura 3-18: Arquitectura red clasificadora

Además de esta red clasificadora, se han entrenado con la salida proporcionada por las redes pre-entrenadas otros modelos, como *random forests*.

3.5 Técnicas para abordar el sobreajuste

Para intentar que no se produzca sobreajuste a la hora de entrenar la red clasificadora se han utilizado varias técnicas. Por un lado, en dicha red se ha empleado *dropout*, en concreto de 0.5, y regularización L2 en las capas densas. Además, como se ha explicado anteriormente se ha realizado *data augmentation* con las imágenes de entrada a las redes pre-entrenadas.

3.6 Software utilizado

El lenguaje de programación escogido, a pesar de contar con un mayor conocimiento del uso de MATLAB, fue Python debido a que cuenta con una gran cantidad de librerías para Deep Learning. Las librerías Deep Learning empleadas han sido Tensorflow y Keras.

4 Integración, pruebas y resultados

Para evaluar los diferentes tipos de preprocesado de las imágenes, así como las arquitecturas y demás técnicas expuestas anteriormente, se han diseñado una serie de pruebas que se describen en esta sección. Además, se expone el método de evaluación empleado para ello y los resultados obtenidos.

4.1 Método de evaluación

Para la evaluación de las distintas pruebas se ha utilizado el área bajo la curva ROC (AUC) de la clasificación final obtenida del sistema de clasificación. La curva ROC (*Receiver Operating Characteristic*) consiste en una representación gráfica de los verdaderos positivos (*true positive rate*) frente a los falsos positivos (*false positive rate*), donde:

$$tpr = \frac{true\ positive}{positive} = \frac{true\ positive}{true\ positive + false\ negative}$$

$$fpr = \frac{false\ positive}{negative} = \frac{false\ positive}{false\ positive + true\ negative}$$

Para cada una de las pruebas realizadas se halla la curva ROC de cada una de las clases respecto al resto y se calcula la media de todas ellas, cuya AUC será la métrica empleada en la comparación.

4.2 Pruebas y resultados

A modo de resumen, la siguiente tabla muestra las distintas posibilidades que existen en cada una de las etapas del sistema de clasificación, detalladas en el capítulo 3 'Metodología', y cuyas combinaciones de distintos modos van a conformar las pruebas realizadas en este trabajo. Mediante la realización de dichas pruebas se busca ver cómo afecta la variación de cada uno de estos parámetros del sistema de clasificación a la predicción final.

Parámetro	Posibles valores
Arquitectura de la red	Inception-v3 VGG16
Tipo de recorte	Ninguno Tipo 1 Tipo 2 Tipo 3
Mejoras de contraste	Ninguna CLAHE Ecualización de histograma Normalización de color

Tamaño de la imagen	256x256 512x512 1024x1024
Balanceo de clases	Ninguno Mediante oversampling (*data augmentation*) Mediante undersampling

Tabla 4-1: Resumen valores de las pruebas

En primer lugar, se realizan una serie de experimentos a fin de comprobar cómo afecta aplicar o no los distintos tipos de preprocesamiento de las imágenes y las distintas técnicas empleadas para ello. En estas pruebas se utiliza siempre la red pre-entrenada Inception-v3 y se van modificando estos parámetros.

Respecto al recorte de las imágenes se quiere comprobar cómo afecta recortar el fondo negro presente en las imágenes que, lógicamente, no aporta ningún tipo de información. Se han realizan cuatro pruebas, una de ellas en la que las imágenes no se recortan y otras tres con cada uno de los tipos de recorte explicados en el capítulo anterior, todas ellas con la red Inception-v3 y fijando el tamaño de imagen a 512×512. En la siguiente tabla se puede observar un resumen de estas pruebas.

Prueba	Recorte
R0	No
R1	1
R2	2
R3	3

Tabla 4-2: Resumen pruebas con distintos tipos de recorte

Tras realizar estos experimentos se comprueba cómo la AUC aumenta ligeramente cuando se realiza un recorte del fondo negro de las imágenes, aunque no difiere mucho entre los tres recortes realizados, siendo mejor la curva ROC para el primero y segundo, cuyas AUC son 0.79, y no variando a penas para el tercero cuya AUC es 0.77, igual que en la prueba donde no se recorta el fondo. Estas curvas se pueden observar en las figuras 5-1 y 5-2.

Por tanto, queda patente que es necesario eliminar el fondo negro lo máximo posible al no aportar nada a la clasificación, ya que la prueba en la que mayor puntación se obtiene es en la que el fondo negro es menor, pero sin eliminar parte del ojo para ello, lo que supone perder parte de la información de la imagen y por tanto se obtiene un peor resultado, como es el caso del tipo de recorte número 3.

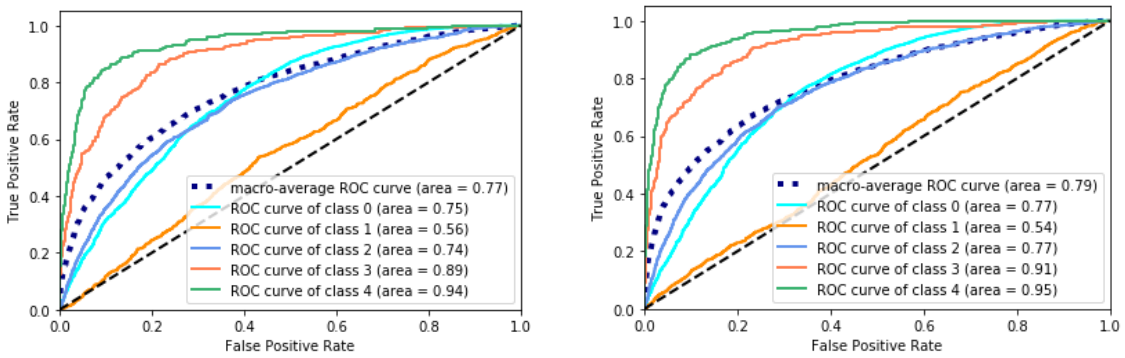


Figura 5-1: Curva ROC pruebas R0 y R1

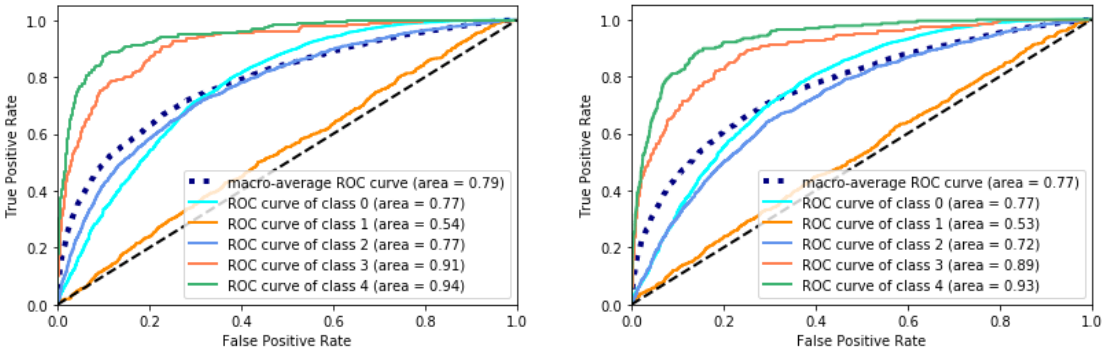


Figura 5-2: Curva ROC pruebas R2 y R3

Respecto a las mejoras de contraste, para comprobar si en las pruebas en las que este se mejora se obtiene un resultado más elevado que en las que no y saber cuál de los distintos tipos es mejor, se realizan cinco pruebas, una de ellas, la prueba C0, en la que no se aplica ninguna mejora de contraste, otras tres en las que se realiza cada uno de los dos tipos de ecualización del histograma y la normalización de color explicados en el capítulo anterior, y una última para comprobar en qué medida el color de la imagen aporta algún tipo de información a la clasificación. Todas ellas con la red Inception-v3 y fijando el tamaño de imagen a 512×512. En la tabla 4-3 se observa un resumen de estas pruebas.

Prueba	Contraste
C0	No
C1	CLAHE en color
C2	Ecualización de histograma en color
C3	Normalización de color
C4	CLAHE en blanco y negro

Tabla 4-3: Resumen pruebas con distintos tipos de mejora de contraste

Tras llevar a cabo estas pruebas se comprueba que el resultado también es mejor si se aplica alguna de las mejoras de contraste, en concreto, la que mejor resultado proporciona es la ecualización CLAHE, cuya AUC es 0.79, tras ella está la ecualización no adaptativa con AUC 0.78, mientras que la normalización de color mejora la AUC de algunas clases, pero empeora la de otras respecto a la prueba C0, por lo que la AUC media será, al igual que en la prueba C0, 0.77. Además, en la prueba C4, realizada con imágenes en blanco y negro, se obtiene un resultado que apenas difiere del obtenido en la C1, que es exactamente igual, pero empleando imágenes en color, obteniéndose en ambas un AUC medio de 0.79. Por tanto, se puede concluir que el color de la imagen no es ni mucho menos lo que aporta la información más relevante a nuestra red, por lo que se podría reducir la carga computacional de todo el proceso utilizando imágenes con un solo canal en vez de tres. Todas estas curvas ROC se pueden observar en las figuras 5-3, 5-4 y 5-5.

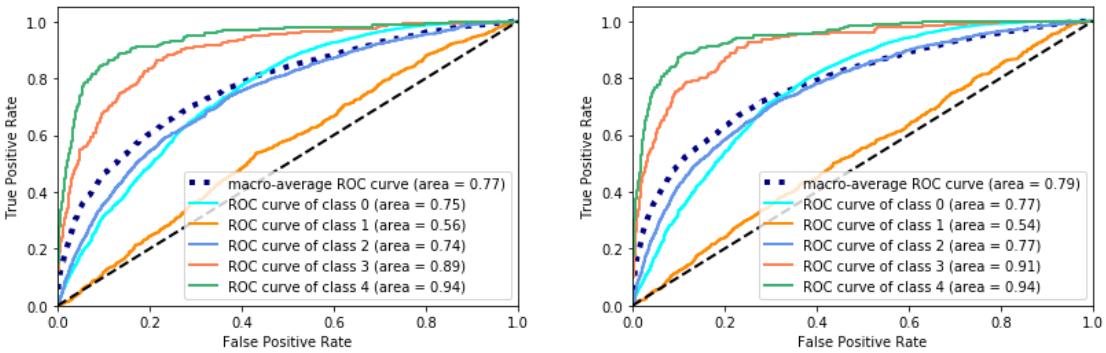


Figura 5-3: Curva ROC pruebas C0 y C1

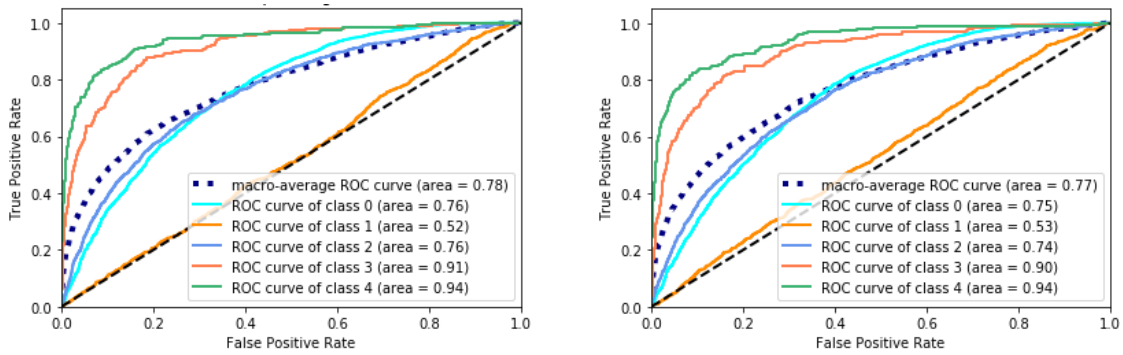


Figura 5-4: Curva ROC pruebas C2 y C3

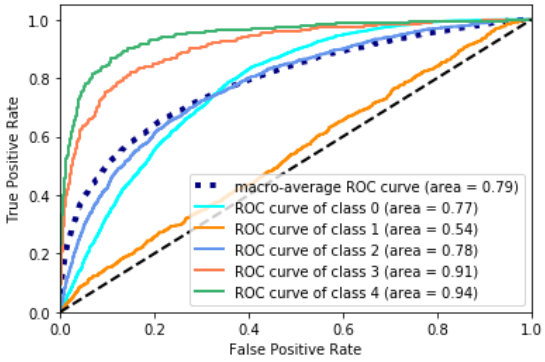


Figura 5-5: Curva ROC prueba C4

Respecto al tamaño de entrada de las imágenes a las redes, este suele oscilar en la literatura consultada entre 256×256 y 512×512. Para comprobar si un mayor detalle en las imágenes facilita la extracción de características, se han realizado 3 experimentos normalizando el tamaño de imagen a distintos valores. En la prueba T1 a 256×256, ya que la mayoría de los trabajos consultados utilizan tamaños de entrada de imagen entorno a este valor, 512×512 para la prueba T2, ya que en algunos trabajos consultados se alcanza este valor, y finalmente 1024×1024 para comprobar si sobrepasar estas dimensiones, utilizadas en la literatura consultada, aporta algún beneficio. En estas tres pruebas, explicadas resumidamente en la tabla 4-4, se utiliza siempre la red Inception-v3 y se fijan el tipo de recorte y normalización de color de las imágenes.

Prueba	Tamaño de imagen
T1	256x256
T2	512x512
T3	1024x1024

Tabla 4-4: Resumen pruebas con distintos tamaños de imagen

Después de la realización de estas pruebas se observa que la AUC es mayor cuanto mayor es el tamaño de entrada de las imágenes a la red, pero mientras que la diferencia entre aquellas de tamaño 256×256 y 512×512 es bastante notoria (la AUC se incrementa de 0.74 a 0.79), al aumentar de 512×512 a 1024×1024 la diferencia ya es bastante menor, solo de 0.79 a 0.8, aunque el coste computacional es notablemente mayor. Por tanto, se concluye que el nivel de detalle que se requiere para la extracción de características es el presente en las imágenes con dimensiones alrededor de 512×512, ya que empleando menores dimensiones se pierde parte de la información, y aumentando las mismas apenas se muestra diferencia siendo la carga computacional del proceso mucho mayor. En las figuras 5-6 y 5-7 se encuentran las curvas ROC pertenecientes a estas pruebas.

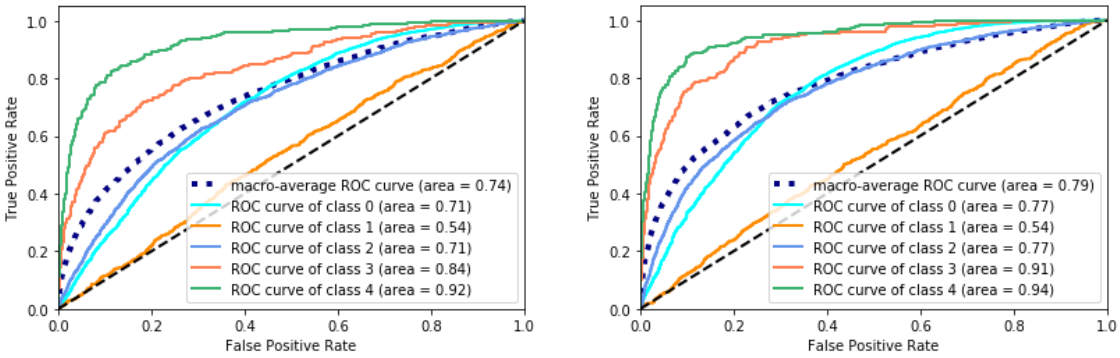


Figura 5-6: Curva ROC pruebas T1 y T2

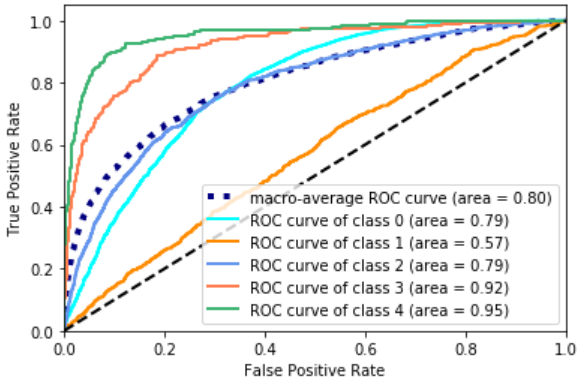


Figura 5-7: Curva ROC prueba T3

Por otro lado, al contar con un *dataset* con clases bastante desbalanceadas, se realizarán también una serie de pruebas para comprobar si la corrección de este desbalanceo mejora la predicción final. Se realizan las pruebas descritas en la tabla 4-5: la B0 y B1 para comprobar si el resultado mejora con una base de datos más balanceada simplemente reduciendo la clase mayoritaria, con tamaño de entrada de las imágenes (256×256), y la B0b y B2 para comprobar qué ocurre si se aplica *data augmentation*, técnica aplicada en la mayoría de la literatura consultada ya que también ayuda a reducir el sobreajuste. En este caso se utiliza un tamaño de entrada de las imágenes de 512×512.

Prueba	Balanceo de clases	Tamaño
B0	No	256x256
B1	Sí (*undersampling*)	256x256
B0b	No	512x512
B2	Sí (*oversampling*)	512x512

Tabla 4-5: Resumen pruebas de balaceo de clases

Mediante los resultados de estas pruebas se observa que, al intentar corregir el alto desbalanceo que existe entre las clases, reducir las clases mayoritarias para igualarlas empeora el resultado, siendo el AUC de la prueba B0 0.74 y el de la prueba B1 0.73, por lo

que reducir aún más las clases para tener una distribución uniforme hará que contemos con un menor número de imágenes para entrenar la red y obtengamos un AUC aún menor. Con la aplicación de *data augmentation* se observa cierta mejoría, pero esta es mucho menor de lo esperado ya que solo se incrementa en ciertas clases, por lo que la media de todas ellas es de 0.79 tanto en la prueba a la que no se le aplica como en la que sí. Por otro lado, es una técnica que implica un gran aumento del coste computacional. Las curvas de estos experimentos se pueden encontrar en las figuras 5-8 y 5-9.

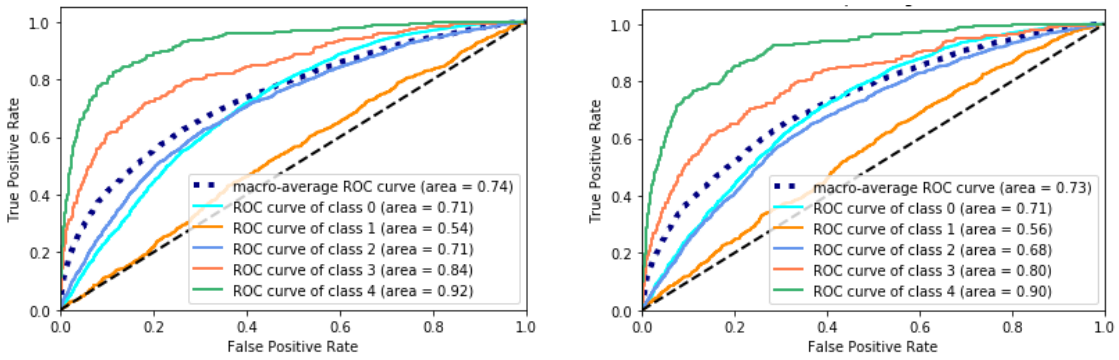


Figura 5-8: Curva ROC pruebas B0 y B1

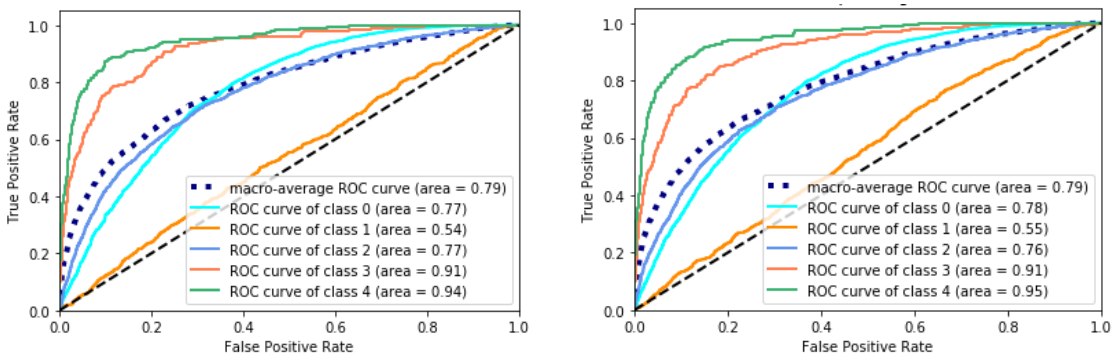


Figura 5-9: Curva ROC pruebas B0b y B2

Por último, como se ha explicado anteriormente, ante la carencia de los medios adecuados para entrenar una red desde cero, se han seleccionado las redes pre-entrenadas Inception-v3 y VGG16 para la extracción de características de la base de datos. De esta manera se podrá observar qué influencia tiene la profundidad de la red en dicha extracción de características, ya que la primera cuenta con bastantes más capas que la segunda. Para ello, se realizaron una serie de pruebas con estas dos redes fijando el resto de los parámetros como se muestra en la siguiente tabla.

Prueba	Arquitectura	Preprocesamiento imágenes		
		Tamaño	Contraste	Recorte
R1a	Inception v3	256x256	Clahe color	2
R2a	VGG16	256 x256	Clahe color	2

R1b	Inception v3	512x512	Clahe color	2
R2b	VGG16	512 x512	Clahe color	2
R1c	Inception v3	1024x1024	Clahe color	2
R2c	VGG16	1024 x1024	Clahe color	2

Tabla 4-6: Resumen pruebas Inception-v3 y VGG16

Tras la realización de estos experimentos se observa que la arquitectura que proporciona mejores resultados es, para todos los tamaños de imagen, Inception-v3, ya que para los mismos preprocesamientos se observan grandes diferencias entre su ROC y la obtenida con VGG16. Un ejemplo de esto serían las pruebas R1b y R2b, cuyas curvas se ven en la figura 5-11, donde el AUC para Inception-v3 es de 0.79 y para VGG16 de 0.63. Esto puede ser debido a que el hecho de entrenar el modelo clasificador con la salida de esta última red, la cual tiene un número de datos cuatro veces menor que Inception-v3, produzca *underfitting*. Las curvas ROC de estas seis pruebas se observan en las siguientes figuras.

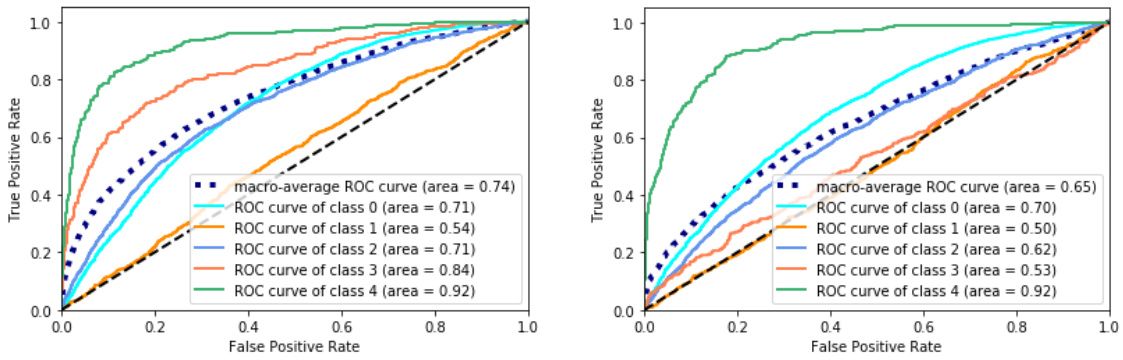


Figura 5-10: Curvas ROC pruebas R1a y R2a

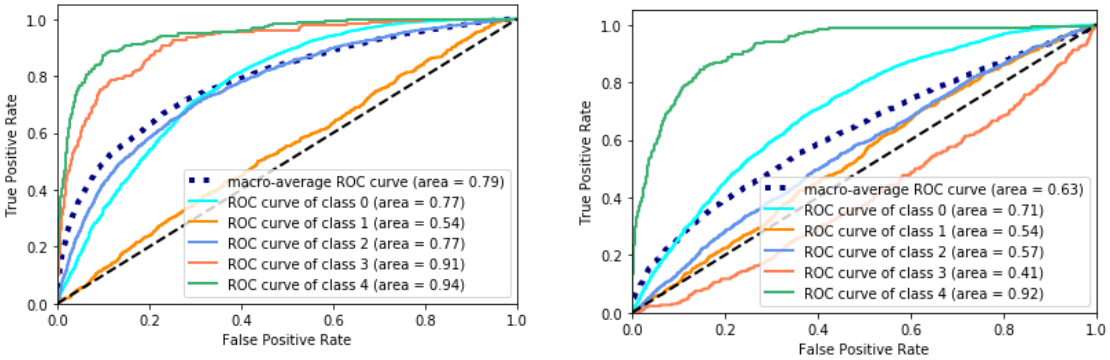


Figura 5-11: Curvas ROC pruebas R1b y R2b

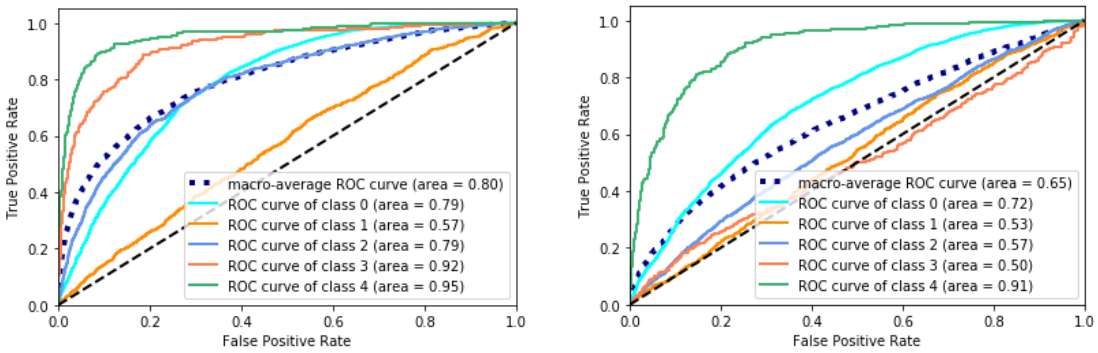


Figura 5-12: Curvas ROC pruebas R1c y R2c

Además, se realizaron dos pruebas con los dos tipos de *ensembles* observados en la fase de estudio del estado del arte: combinando varias salidas de la misma red, y combinando salidas de distintas redes pre-entrenadas. En concreto, el primero de ellos combinando las salidas de Inception-v3 y VGG16 con las mismas imágenes de entrada, y el segundo combinando dos salidas de Inception-v3 con imágenes de entradas procesadas de distinta forma, como muestra la siguiente tabla.

Prueba	Arquitectura	Preprocesamiento imágenes		
		Tamaño	Contraste	Recorte
E1	Ensemble Inception v3 y VGG16	512x512	Clahe color	2
E2	Inception v3	512x512	Clahe color	Ensemble 1 y 2

Tabla 4-7: Resumen pruebas *ensembles*

Mediante los *ensembles* realizados, se comprueba que si bien combinando diferentes salidas de la misma red no mejora el resultado (la AUC sigue siendo 0.79, igual que la de las dos pruebas que se combinan por separado), combinando las salidas de las dos redes este sí que mejora, teniendo una AUC de 0.81, que es superior a todas las obtenidas anteriormente. Por ello se concluye que, aunque por separado no den resultados muy buenos como es el caso de VGG16, combinar las salidas de distintas redes mejora la AUC. En la figura 5-13 se encuentran las curvas ROC obtenidas de los *ensembles*.

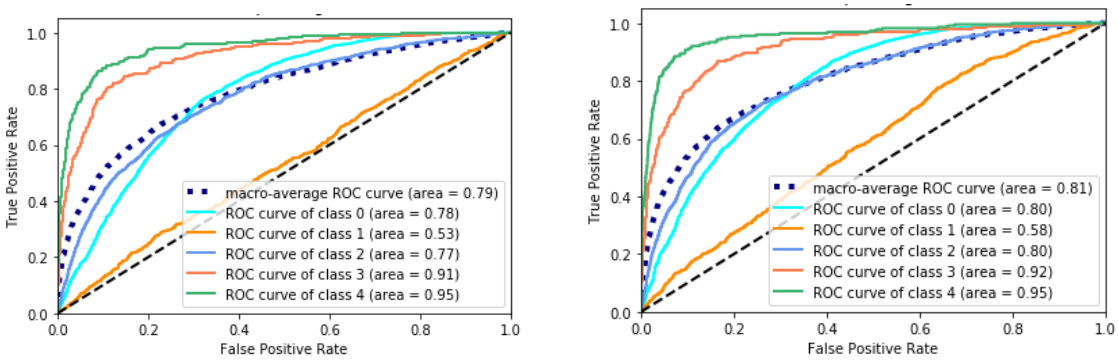


Figura 5-13: Curva ROC pruebas E1 y E2

Además, hay que comentar que el score obtenido por la red clasificadora es, en todos los casos, mayor que el de los otros modelos entrenados con la salida de las redes pre-entrenadas, como los *random forests.*

Llegado a este punto, es evidente que en todas las pruebas se obtiene un resultado demasiado bajo para la clase 1, lo cual hace que la media descienda notablemente y se obtenga un resultado bastante alejado de los hallados mediante el estudio del estado del arte, los cuales suelen superar el 0.9 en su AUC. Por ello, se ha decidido juntar las clases 0 y 1 en una sola, ya que esta última pertenece a imágenes de retinas que presentan la enfermedad en sus primeras fases donde esta es casi imperceptible.

A continuación, se muestran las curvas ROC de algunas de las pruebas explicadas en este apartado, halladas para estas cuatro nuevas clases:

Con estas nuevas clases, la diferencia entre la prueba en la que no se mejora el contraste y aquella en la que se aplicaba la ecualización CLAHE es algo más notoria, y lo mismo ocurre entre la que no se recortan las imágenes (prueba R0) y en la que se aplica el tipo de recorte 1 (prueba R1). Como se observa en la figura 5-14 esta aumenta de 0.83 a 0.87.

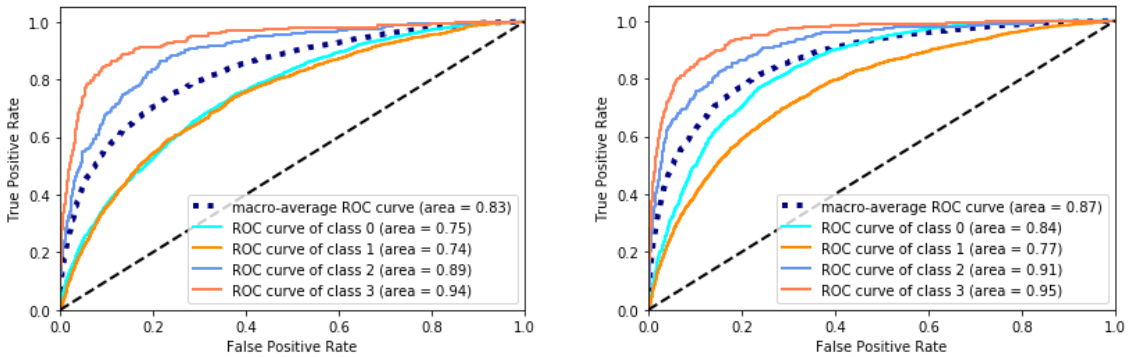


Figura 5-14: Curva ROC pruebas R0 y R1 para 4 clases

Por otro lado, en las pruebas donde se comparan los distintos tamaños de imagen, la mejora al aumentar la resolución de 256×256 a 512×512, cuyas curvas se muestran en las siguientes figuras, es aún más evidente, hasta 7 puntos cuando para cinco clases eran solo

5, mientras que la diferencia al aumentar de 512×512 a 1024×1024 sigue siendo muy pequeña, tan solo de un punto.

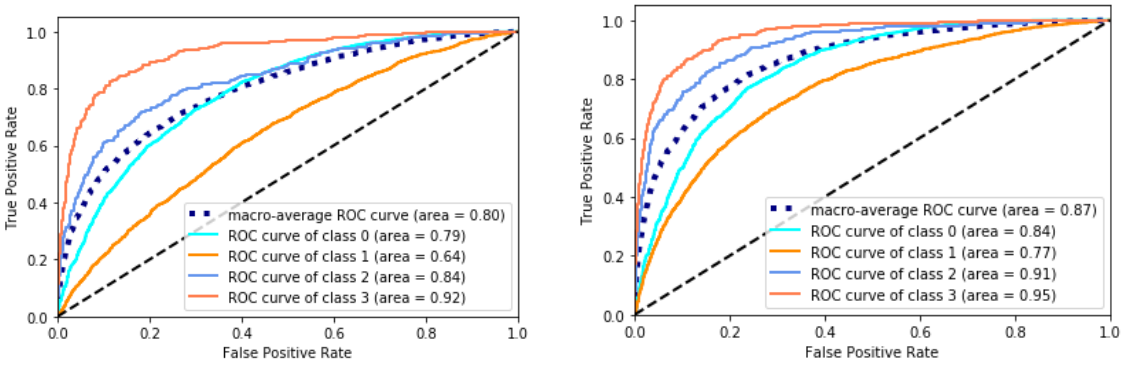


Figura 5-15: Curva ROC pruebas T1 y T2 para 4 clases

Para las dos redes empleadas, Inception-v3 y VGG16, ocurre lo mismo, la curva ROC mejora bastante al hallarla para estas cuatro nuevas clases, aunque la diferencia entre ellas sigue siendo similar. Y finalmente, en la prueba donde se obtenía una mayor puntuación, el *ensemble* formado por las salidas de estas dos redes, al tomar estas dos clases como una sola, el AUC medio aumenta hasta 0.88, valor ya más cercano a los resultados hallados en el estado del arte. En la figura 5-16 se puede observar su curva ROC.

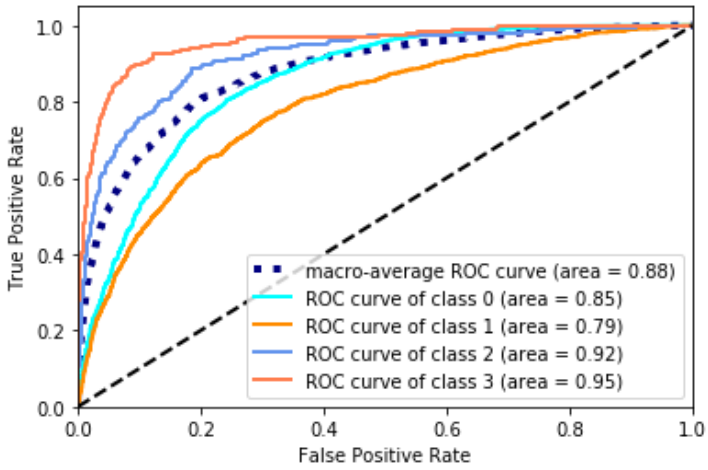


Figura 5-16: Curva ROC prueba E1 para 4 clases

Por tanto, queda claro que al tomar las clases 0 y 1 como una sola, la ROC para todas las pruebas mejora en gran medida, lo cual es debido a la mínima diferencia existente entre estas dos clases, casi imperceptible también para el ojo humano. Al extraer de ellas características mediante una red entrenada con imágenes de otro tipo, esta no es tan buena extrayendo los pequeños detalles que se requieren para diferenciar imágenes de dos clases tan parecidas entre sí.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

El objetivo de este trabajo consiste en llevar a cabo una comparación del rendimiento de diferentes métodos a fin de exponer cuáles de ellos resultan de mayor o menor ayuda a la hora de enfrentarse a un problema de clasificación de imagen médica mediante Deep Learning.

Tras un estudio en profundidad del estado del arte se han hallado distintas técnicas para el preprocesado de las imágenes que componen la base de datos, así como varias redes mediante las cuales extraer características de estas. De entre todas ellas, se han seleccionado algunas para aplicar en este trabajo de fin de grado en particular. En concreto, se han escogido tres tipos de recorte y redimensionado de las imágenes y otros tres de mejoras de contraste, así como varios tamaños de imagen con los que realizar los experimentos. También se han probado dos técnicas para balancear las clases dentro del dataset y, ante la imposibilidad de entrenar redes neuronales profundas desde 0, se han realizado los experimentos con dos redes pre-entrenadas. Además, se han probado ciertas técnicas para tratar de reducir el sobreajuste.

De todos los experimentos realizados se ha concluido que el fondo negro de las imágenes es ruido, por lo que cuanto más se elimine sin perder información útil mejores resultados se obtendrán, que aplicar una mejora del contraste a las imágenes, y en concreto la ecualización adaptativa CLAHE, también mejora el resultado, y que las dimensiones de entrada de las imágenes a la red óptimo para este problema es alrededor de los 512×512 píxeles. Además, al tratar de balancear las clases se ha observado que hacerlo eliminando imágenes no ayuda, ya que cuanto mayor sea nuestro *dataset* el entrenamiento será mejor, y que la técnica del *data augmentation* no ha proporcionado los resultados esperados, por lo que habrá que tratar de perfeccionar la forma de llevarlo a cabo. Por último, se ha observado que la técnica que da mejores resultados es realizar un *ensemble* de varias redes pre-entrenadas que por separado no tienen necesariamente que dar un resultado tan bueno.

5.2 Trabajo futuro

A la vista de los resultados obtenidos se plantea, como trabajo futuro, profundizar en lo siguiente:

- Desarrollo de técnicas de *data augmentation* que consigan mejorar de forma notable los resultados obtenidos al aplicarlas.

- Continuar la línea de investigación en los *ensembles* de redes pre-entrenadas, ya que, sin la necesidad de entrenar desde cero una red neuronal profunda, consiguen resultados muy buenos.

- Estudiar la posibilidad de la implantación de un sistema de clasificación de imagen médica en sistemas de asistencia médica reales.

Referencias

Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al., (2015): "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems". Disponible en: https://arxiv.org/abs/1603.04467

Advanced Guide to Inception v3 on Cloud TPU | Cloud TPU | Google Cloud [Internet]. [citado 18 de mayo de 2019]. Disponible en: https://cloud.google.com/tpu/docs/inception-v3-advanced

Antony J, McGuinness K, Connor NEO, Moran K, (2016): "Quantifying Radiographic Knee Osteoarthritis Severity using Deep Convolutional Neural Networks". arXiv:160902469 [cs]. Disponible en: http://arxiv.org/abs/1609.02469

Antony M, Brüggemann S, Kong H, (2015): "Kaggle Diabetic Retinopathy Detection Team o_O solution". Disponible En: https://www.kaggle.com/blobs/download/forum-message-attachment-files/2797/report.pdf

Berger T, (2015): Machine Learning project: "Kaggle competition for Diabetic Retinopathy detection": teberger/diabetic-retinopathy [Internet]. Disponible en: https://github.com/teberger/diabetic-retinopathy

Bermudez I. El Sistema Nervioso: Biología [Internet]. El Sistema Nervioso. 2011 [citado 18 de mayo de 2019]. Disponible en: http://elsistemanervioso-uam.blogspot.com/2011/04/biologia.html

Bahrampour S, Ramakrishnan N, Schott L, Shah M, (2015): "Comparative Study of Deep Learning Software Frameworks". arXiv:151106435 [cs] [Internet]. 19 de noviembre de 2015; Disponible en: http://arxiv.org/abs/1511.06435

Chase G, (2017): "Identifying diabetic retinopathy using convolutional neural networks": gregwchase/dsi-capstone [Internet]. Disponible en: https://github.com/gregwchase/dsi-capstone

Chen X, Xu Y, Kee Wong DW, Wong TY, Liu J, (2015): "Glaucoma detection based on deep convolutional neural network", En: 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). Milan: IEEE; 2015. p. 715-8. Disponible en: https://ieeexplore.ieee.org/document/7318462/

Ciresan D, Giusti A, Gambardella LM, Schmidhuber J, (2012): "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images". In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'12), pp. 2843-2851. Disponible en: http://people.idsia.ch/~juergen/nips2012.pdf

Collobert R, Kavukcuoglu K, Farabet C, (2011): Torch7: "A Matlab-like Environment for Machine Learning". Disponible en: https://pdfs.semanticscholar.org/3449/b65008b27f6e60a73d80c1fd990f0481126b.pdf?_ga=2.167632190.167280239.1558191360-2014392946.1558191360

CS231n Convolutional Neural Networks for Visual Recognition [Internet]. Disponible en: http://cs231n.github.io/convolutional-networks/

Deep Learning Toolbox [Internet]. Disponible en: https://es.mathworks.com/products/deep-learning.html

Devillers L, Vidrascu L, Lamel L, (2005): "Challenges in real-life emotion annotation and machine learning based detection. Neural Networks". Vol.18(4):407-22. Disponible en: https://www.sciencedirect.com/science/article/pii/S0893608005000407

Esteva A, Kuprel B, Novoa RA, Ko J, Swetter SM, Blau HM, et al., (2017): "Dermatologist-level classification of skin cancer with deep neural networks". Nature 542(7639):115-8. Disponible en: https://www.ncbi.nlm.nih.gov/pubmed/28117445

Graham B, (2015): "Kaggle Diabetic Retinopathy Detection competition report". Disponible en: https://kaggle2.blob.core.windows.net/forum-message-attachments/88655/2795/competitionreport.pdf

Gulshan V, Peng L, Coram M, Stumpe MC, Wu D, Narayanaswamy A, et al., (2016): "Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs". Journal of the American Medical Association 316(22):2402. Disponible en: https://jamanetwork.com/journals/jama/fullarticle/2588763

Ioffe S, Szegedy C, (2015): "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". arXiv:150203167 [cs] [Internet]. Disponible en: http://arxiv.org/abs/1502.03167

Kaggle Diabetic Retinopathy Detection [Internet]. Disponible en: https://www.kaggle.com/c/diabetic-retinopathy-detection

Kawahara J, Hamarneh G, (2016): "Multi-resolution-Tract CNN with Hybrid Pretrained and Skin-Lesion Trained Layers". En: Wang L, Adeli E, Wang Q, Shi Y, Suk H-I, editores. Machine Learning in Medical Imaging [Internet]. Cham: Springer International Publishing; 2016. p. 164-71. Disponible en: http://link.springer.com/10.1007/978-3-319-47157-0_20

Keras Documentation [Internet]. Disponible en: https://keras.io/#why-this-name-keras

Krizhevsky A, Sutskever I, Hinton GE, (2012): "ImageNet classification with deep convolutional neural networks". Neural Information Processing Systems. 25. 10.1145/3065386. Disponible en: https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

LeCun Y, Bengio Y, Hinton G, (2015): "Deep learning". Nature vol. 521, pp.436-44. Disponible en: https://www.nature.com/articles/nature14539

Litjens G, Kooi T, Bejnordi BE, Setio AAA, Ciompi F, Ghafoorian M, et al., (2017): "A Survey on Deep Learning in Medical Image Analysis". Medical Image Analysis vol. 42:60-88. Disponible en: https://www.sciencedirect.com/science/article/abs/pii/S1361841517301135?via%3Dihub

Microsoft releases beta of Microsoft Cognitive Toolkit for deep learning advances [Internet]. The AI Blog. 2016. Disponible en: https://blogs.microsoft.com/ai/microsoft-releases-beta-microsoft-cognitive-toolkit-deep-learning-advances/

Murray N, Perronnin F, (2014): "Generalized Max Pooling". En: 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA: IEEE; 2014. p. 2473-80. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909713

OpenCV: Histograms - 2: Histogram Equalization [Internet]. Disponible en: https://docs.opencv.org/3.1.0/d5/daf/tutorial_py_histogram_equalization.html

Parkhi OM, Vedaldi A, Zisserman A, (2015): "Deep Face Recognition". En: Procedings of the British Machine Vision Conference 2015 [Internet]. Swansea: British Machine Vision Association; 2015. p. 41.1-41.12. Disponible en: http://www.bmva.org/bmvc/2015/papers/paper041/index.html

Perceptrón multicapa. En: Wikipedia, la enciclopedia libre [Internet]. 2018 [citado 18 de mayo de 2019]. Disponible en: https://es.wikipedia.org/w/index.php?title=Perceptr%C3%B3n_multicapa&oldid=108448948

Pratt H, Coenen F, Broadbent DM, Harding SP, Zheng Y, (2016): "Convolutional Neural Networks for Diabetic Retinopathy". Procedia Computer Science. 2016; 90:200-5. Disponible en: https://core.ac.uk/download/pdf/80776196.pdf

Rasta S, Eisazadeh Partovi M, Seyedarabi H, Javadzadeh A, (2015): "A Comparative Study on Preprocessing Techniques in Diabetic Retinopathy Retinal Images: Illumination Correction and Contrast Enhancement". Journal of Medical Signals and Sensors vol 5:40-8. Disponible en: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4335144/

Redes Neuronales: una visión superficial - Fernando Sancho Caparrini [Internet]. [citado 18 de mayo de 2019]. Disponible en: http://www.cs.us.es/~fsancho/?e=72

Rectificador (redes neuronales). En: Wikipedia, la enciclopedia libre [Internet]. 2019 [citado 18 de mayo de 2019]. Disponible en: https://es.wikipedia.org/w/index.php?title=Rectificador_(redes_neuronales)&oldid=115409078

Ronneberger, O., Fischer, P., Brox, T., (2015): "U-net: Convolutional networks for biomedical image segmentation". In: Med Image Comput Comput Assist Interv vol. 9351 of Lect Notes Comput Sci. pp. 234–241. Disponible en: https://arxiv.org/abs/1505.04597

Ruder S, (2016): "An overview of gradient descent optimization algorithms". arXiv:160904747 [cs] [Internet]. Disponible en: http://arxiv.org/abs/1609.04747

Schwing AG, Urtasun R, (2015): "Fully Connected Deep Structured Networks." arXiv:150302351; Disponible en: http://arxiv.org/abs/1503.02351

SHARMA S. Activation Functions in Neural Networks [Internet]. Towards Data Science. 2017 [citado 18 de mayo de 2019]. Disponible en: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Shen D, Wu G, Suk H-I, (2017): "Deep Learning in Medical Image Analysis." Annual Review of Biomedical Engineering 19(1):221-48. Disponible en: https://koreauniv.pure.elsevier.com/en/publications/deep-learning-in-medical-image-analysis

Shen, W., Zhou, M., Yang, F., Yang, C., Tian, J., (2015b): "Multi-scale convolutional neural networks for lung nodule classification". In: Inf Process Med Imaging vol. 9123 of Lect Notes Comput Sci. pp. 588–599. Disponible en: https://www.ncbi.nlm.nih.gov/pubmed/26221705

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R, (2014): "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". Disponible en: http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z, (2016): "Rethinking the Inception Architecture for Computer Vision". En: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [Internet]. Las Vegas, NV, USA: IEEE; 2016. p. 2818-26. Disponible en: http://ieeexplore.ieee.org/document/7780677/

Theano at a Glance — Theano 1.0.0 documentation [Internet]. Disponible en: http://deeplearning.net/software/theano/introduction.html

Ting DSW, Cheung CY-L, Lim G, Tan GSW, Quang ND, Gan A, et al., (2017): "Development and Validation of a Deep Learning System for Diabetic Retinopathy and Related Eye Diseases Using Retinal Images from Multiethnic Populations with Diabetes". Journal of the American Medical Association 318(22):2211. Disponible en: https://www.ncbi.nlm.nih.gov/pubmed/29234807

Van den Oord A, Dieleman S, Schrauwen B, (2013): "Deep content-based music recommendation". Diponible en: https://papers.nips.cc/paper/5004-deep-content-based-music-recommendation

VGG16 - Convolutional Network for Classification and Detection [Internet]. 2018. Disponible en: https://neurohive.io/en/popular-networks/vgg16/

Worrall DE, Wilson CM, Brostow GJ, (2016): "Automated Retinopathy of Prematurity Case Detection with Convolutional Neural Networks". En: Carneiro G, Mateus D, Peter L, Bradley A, Tavares JMRS, Belagiannis V, et al., editores. Deep Learning and Data Labeling for Medical Applications [Internet]. Cham: Springer International Publishing; 2016. p. 68-76. Disponible en: http://link.springer.com/10.1007/978-3-319-46976-8_8

Xin Yao, (1999): "Evolving artificial neural networks". Proceedings of the IEEE 87(9):1423-47. Disponible en: http://avellano.fis.usal.es/~lalonso/compt_soft/articulos/yao99evolving.pdf

Zhang C, Bengio S, Hardt M, Recht B, Vinyals O, (2016): “Understanding deep learning requires rethinking generalization”. arXiv:161103530 [cs] [Internet]. Disponible en: http://arxiv.org/abs/1611.03530